AFRL-RH-BR-TR-2008-0006

BTEC Thermal Model

Lance J. Irvin
P.D.S. Maseberg
Gavin D. Buffington

Fort Hays State University

Clifton D. Clark III

Northrop Grumman Information Technology

Robert J. Thomas
Michael L. Edwards
Jacob Stolarski

Air Force Research Laboratory

October 2007
Interim Report

Air Force Research Laboratory
Human Effectiveness Directorate
Directed Energy Bioeffects Division
Optical Radiation Branch
Brooks-City-Base, TX 78235

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the Air Force Research Laboratory, Brooks City-Base, Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RH-BR-TR-2008-0006 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

//SIGNED// _____
ROBERT J. THOMAS, DR-IV
Contract Monitor

//SIGNED// _____
GARRETT D. POLHAMUS, DR-IV, DAF
Chief, Directed Energy Bioeffects Division

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
October 2007	Interim Technical Report	November 2003- October 2007

4. TITLE AND SUBTITLE

BTEC Thermal Model

5a. CONTRACT NUMBER
F41624-02-D-7003

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62202F

6. AUTHOR(S)
[+]Irvin, Lance J.; [+]Maseberg, P.D.S.; [+]Buffington, Gavin D.; [□]Clark III, Clifton D.
[◊]Thomas, Robert J.; [◊]Edwards, Michael L.; [◊]Stolarski, Jacob

5d. PROJECT NUMBER
7757

5e. TASK NUMBER
B2

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

[+]Fort Hays State University
Dept of Physics
600 Park Street
Hays KS, 67601

[◊]Air Force Research Laboratory
Human Effectiveness Directorate
Directed Energy Bioeffects Division
2624 Louis Bauer Dr.
Brooks City-Base, TX 78235-5128

[□]Northrop Grumman Corporation
Information Technology
4241 Woodcock Drive, Ste B-100
San Antonio, TX 78228

5f. WORK UNIT NUMBER
26

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Materiel Command
Human Effectiveness Directorate
Directed Energy Bioeffects Division
Optical Radiation Branch
2624 Louis Bauer Dr.
Brooks City-Base, TX 78235-5128

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RH

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-RH-BR-TR-2008-0006

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

Contract Monitor: Dr. Robert. J. Thomas

14. ABSTRACT

AFRL/RHDO has developed a configurable, laser-tissue interaction model that includes components from various areas of Biophysics. The model predicts heat transfer in biological tissue, in either one-dimension or two-dimensional cylindrical coordinates, and is coupled to an Arrhenius damage model. A simulation can be configured as a single run, or a damage-threshold search. Multiple models for describing the laser-tissue interaction are available, including linear absorption (1D, 2D), Monte Carlo scattering (2D) and Beam Propagation Methods using Finite Difference approximations or Hankel Transform methods (2D).

15. SUBJECT TERMS

thermal model, Arrehius integral, laser-tissue interaction, finite-differences

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Robert J. Thomas
Unclassified	Unclassified	Unclassified	SAR	99	19b. TELEPHONE NUMBER (include area code)

Standard Form 298 (Rev. 8-98)

This page intentionally left blank

Contents

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

The BTEC thermal model is a 1-D and 2-D cylindrical coordinate system simulation of optical radiation and radio frequency thermal interaction with tissues. The code supports the illumination of tissues by sources, the temperature response from the linear absorption of optical radiation, and analysis of subsequent damage. The model takes its name from the four primary authors of the model: Buffington, Thomas, Edwards, and Clark, although many others have contributed.

Finite difference numerical methods are employed in the solutions of heat transfer. An alternating-direction-implicit (ADI) finite difference method is employed in the solution of the heat equation in 2-D, while a Crank-Nicholson Method is employed in the 1-D solution. The BTEC simulation can be configured with a source term defined by a single or by multiple emitters, such as laser, RF, or broadband. The BTEC can also compute a source based on the initial electric field profile and a refractive index distribution.

The tissue simulation is represented as a one-dimensional stack of homogeneous layers along the z-axis. Each layer can have differing thermal, optical, and physical properties. The linear absorption coefficient of each layer defines the energy transfer from the optical source to the tissue. Boundary conditions along the axial and radial (2-D) coordinates may be selected as a sink (temperature held constant) or as a combination of surface boundary conditions (convection, radiative, or evaporative).

The BTEC model currently employs a single rate-process model of thermal injury. User-defined parameters for damage rates and activation energies as a function of temperature can be programmed for each tissue type or layer used. A number of damage integral values or temperature shift searches are available in order to estimate damage thresholds or for comparison to experimental data.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 2

One-Dimensional Heat Equation

2.1 Heat Equation

The BTEC model employs a Crank-Nicholson Method [9] for the solution of the time-dependent, one-dimensional heat transfer problem. A detailed derivation of the method is presented below.

The formulation begins with the time-dependent heat equation expressed in equation 2.1:

$$\rho(\mathbf{r})\, c(\mathbf{r}) \frac{\partial \nu(\mathbf{r}, t)}{\partial t} = \nabla \cdot [\kappa(\mathbf{r}, \nu) \nabla \nu(\mathbf{r}, t)] + g(\mathbf{r}, t) \tag{2.1}$$

Here, $\rho(\mathbf{r})$ is the material density, $c(\mathbf{r})$ is the specific heat, and $\kappa(\mathbf{r}, \nu)$ is the thermal conductivity. The expression $\nu(\mathbf{r}, t)$ is the temperature rise and $g(\mathbf{r}, t)$ is the source, or driving term. The source term contains the effect of an optical or radio-frequency source along with the effects of perfusion (blood flow).

The product rule is used to expand this equation, yielding

$$\rho(\mathbf{r})\, c(\mathbf{r}) \frac{\partial \nu(\mathbf{r}, t)}{\partial t} = \nabla \kappa(\mathbf{r}, \nu) \cdot \nabla \nu(\mathbf{r}, t) + \kappa(\mathbf{r}, \nu)\, \nabla^2 \nu(\mathbf{r}, \nu) + g(\mathbf{r}, t). \tag{2.2}$$

equation 2.2 can then be expanded for one-dimensional Cartesian coordinates using the following:

$$\nabla \kappa = \frac{\partial \kappa}{\partial x} \hat{x} \tag{2.3}$$

$$\nabla \nu = \frac{\partial \nu}{\partial x} \hat{x} \tag{2.4}$$

$$\nabla^2 \nu = \frac{\partial^2 \nu}{\partial x^2} \tag{2.5}$$

Utilizing Equations 2.3, 2.4, and 2.5, the expanded heat equation is

$$\rho(x) c(x) \frac{\partial \nu(x, t)}{\partial t} = \frac{\partial \kappa(x, t)}{\partial x} \frac{\partial \nu(x, t)}{\partial x} + \kappa(x, t) \frac{\partial^2 \nu(x, t)}{\partial x^2} + g(x, t). \tag{2.6}$$

2.2 Finite Difference Representation

The expanded heat equation from equation 2.6 is transformed to a finite difference representation. Space is discretized such that calculations are computed from an indexed position, x_i where

$$i = 0, 1, ..., M-2, M-1.$$

This spatial representation yields a finite difference representation:

$$\rho_i c_i \frac{\nu_i^{n+1} - \nu_i^n}{\Delta t} = \delta_x \kappa_i \frac{\delta_x \nu_i^{n+1} + \delta_x \nu_i^n}{2} + \kappa_i \frac{\delta_x^2 \nu_i^{n+1} + \delta_x^2 \nu_i^n}{2} + g_i \tag{2.7}$$

In equation 2.7, the Crank-Nicholson Method is used. The forward finite difference at time t_n and backward finite difference at time t_{n+1} are averaged in the Crank-Nicholson Method.

By multiplying equation 2.7 by 2 the expression

$$\frac{2\rho_i c_i}{\Delta t}\left(\nu_i^{n+1} - \nu_i^n\right) = \delta_x \kappa_i \, \delta_x \nu_i^{n+1} + \delta_x \kappa_i \, \delta_x \nu_i^n + \kappa_i \, \delta_x^2 \nu_i^{n+1} + \kappa_i \, \delta_x^2 \nu_i^n + 2g_i \tag{2.8}$$

is found. Next, to group known and unknown terms, all ν_i^{n+1} terms are moved to the left hand side (LHS) and all ν_i^n terms are moved to the right hand side (RHS):

$$\frac{2\rho_i c_i}{\Delta t}\nu_i^{n+1} - \left(\delta_x \kappa_i \, \delta_x \nu_i^{n+1} + \kappa_i \delta_x^2 \nu_i^{n+1}\right) = \frac{2\rho_i c_i}{\Delta t}\nu_i^n + \delta_x \kappa_i \, \delta_x \nu_i^n + \kappa_i \, \delta_x^2 \nu_i^n + 2g_i \tag{2.9}$$

The derivative operators on ν are expanded for final simplification using stretchable finite differences. Stretchable finite-derivatives are used if a non-uniformly spaced grid is desired. Finite differences are used to represent continuous derivatives in discrete computational space. Table 2.1 shows finite difference operators that will be used. Due to the size of the equations, the calculations are presented in two parts, the LHS and the RHS.

Operator	Un-stretched	Stretched
$\delta_x f_i$	$\frac{f_{i+1} - f_{i-1}}{2h}$	$\frac{\Delta x_{i-}}{\Delta x_i \Delta x_{i+}} f_{i+1} + \frac{\Delta x_{i+} - \Delta x_{i-}}{\Delta x_{i-} \Delta x_{i+}} f_i - \frac{\Delta x_{i+}}{\Delta x_i \Delta x_{i-}} f_{i-1}$
$\delta_x^2 f_i$	$\frac{f_{i+1} - 2f_i + f_{i-1}}{h^2}$	$\frac{2}{\Delta x_i \Delta x_{i+}} f_{i+1} - \frac{2}{\Delta x_{i+} \Delta x_{i-}} f_i + \frac{2}{\Delta x_i \Delta x_{i-}} f_{i-1}$

Table 2.1: Finite Difference Operator Table for 1-D Heat Equation

Here h is a constant step size between points and the Δx's are defined as

$$\Delta x_i = x_{i+1} - x_{i-1}$$

$$\Delta x_{i+} = x_{i+1} - x_i$$

$$\Delta x_{i-} = x_i - x_{i-1}.$$

Left-Hand Side of 1-D Finite Difference Form

The LHS is defined as

$$LHS_i = \frac{2\rho_i c_i}{\Delta t}\nu_i^{n+1} - \left(\delta_x \kappa_i\, \delta_x \nu_i^{n+1} + \kappa_i \delta_x^2 \nu_i^{n+1}\right). \tag{2.10}$$

Expanding the operators on ν,

$$\begin{aligned} LHS_i = & \frac{2\rho_i c_i}{\Delta t}\nu_i^{n+1} \\ & - \delta_x \kappa_i \left[\frac{\Delta x_{i-}}{\Delta x_i \Delta x_{i+}}\nu_{i+1}^{n+1} + \frac{\Delta x_{i+} - \Delta x_{i-}}{\Delta x_{i-}\Delta x_{i+}}\nu_i^{n+1} - \frac{\Delta x_{i+}}{\Delta x_i \Delta x_{i-}}\nu_{i-1}^{n+1}\right] \\ & - \kappa_i \left[\frac{2}{\Delta x_i \Delta x_{i+}}\nu_{i+1}^{n+1} - \frac{2}{\Delta x_{i-}\Delta x_{i+}}\nu_i^{n+1} + \frac{2}{\Delta x_i \Delta x_{i-}}\nu_{i-1}^{n+1}\right] \end{aligned} \tag{2.11}$$

and collecting similar ν terms, the expression

$$\begin{aligned} LHS_i = & -\frac{1}{\Delta x_i \Delta x_{i+}}\left[2\kappa_i + \delta_x \kappa_i \Delta x_{i-}\right]\nu_{i+1}^{n+1} \\ & + \left[\frac{2\rho_i c_i}{\Delta t} - \frac{\delta_x \kappa_i [\Delta x_{i+} - \Delta x_{i-}] - 2\kappa_i}{\Delta x_{i-}\Delta x_{i+}}\right]\nu_i^{n+1} \\ & - \frac{1}{\Delta x_i \Delta x_{i-}}\left[2\kappa_i - \delta_x \kappa_i \Delta x_{i+}\right]\nu_{i-1}^{n+1} \end{aligned} \tag{2.12}$$

is found. Now, if the following constants are defined,

$$a_i^L = -\frac{1}{\Delta x_i \Delta x_{i-}}\left[2\kappa_i - \delta_x \kappa_i \Delta x_{i+}\right] \tag{2.13}$$

$$b_i^L = \frac{2\rho_i c_i}{\Delta t} - \frac{\delta_x \kappa_i [\Delta x_{i+} - \Delta x_{i-}] - 2\kappa_i}{\Delta x_{i-}\Delta x_{i+}} \tag{2.14}$$

$$c_i^L = -\frac{1}{\Delta x_i \Delta x_{i+}}\left[2\kappa_i + \delta_x \kappa_i \Delta x_{i-}\right] \tag{2.15}$$

the LHS can be represented in a simplified format:

$$LHS_i = a_i^L \nu_{i-1}^{n+1} + b_i^L \nu_i^{n+1} + c_i^L \nu_{i+1}^{n+1} \tag{2.16}$$

Right-Hand Side of 1-D Finite Difference Form

Following the same formulation as for the LHS, the RHS becomes

$$RHS_i = \frac{2\rho_i c_i}{\Delta t}\nu_i^n + \delta_x \kappa_i\, \delta_x \nu_i^n + \kappa_i\, \delta_x^2 \nu_i^n + 2g_i. \tag{2.17}$$

The operators on ν are expanded to form

$$\begin{aligned} RHS_i = & \frac{2\rho_i c_i}{\Delta t}\nu_i^n \\ & \delta_x \kappa_i \left[\frac{\Delta x_{i-}}{\Delta x_i \Delta x_{i+}}\nu_{i+1}^n + \frac{\Delta x_{i+} - \Delta x_{i-}}{\Delta x_{i-}\Delta x_{i+}}\nu_i^n - \frac{\Delta x_{i+}}{\Delta x_i \Delta x_{i-}}\nu_{i-1}^n\right] \\ & + \kappa_i \left[\frac{2}{\Delta x_i \Delta x_{i+}}\nu_{i+1}^n - \frac{2}{\Delta x_{i-}\Delta x_{i+}}\nu_i^n + \frac{2}{\Delta x_i \Delta x_{i-}}\nu_{i-1}^n\right] \\ & + 2g_i \end{aligned} \tag{2.18}$$

and then by collecting similar ν terms, the expression

$$\begin{aligned} RHS_i = & \frac{1}{\Delta x_i \Delta x_{i+}} \left[2\kappa_i + \delta_x \kappa_i \Delta x_{i-}\right] \nu_{i+1}^n \\ & + \left[\frac{2\rho_i c_i}{\Delta t} + \frac{\delta_x \kappa_i [\Delta x_{i+} - \Delta x_{i-}] - 2\kappa_i}{\Delta x_{i-} \Delta x_{i+}}\right] \nu_i^n \\ & + \frac{1}{\Delta x_i \Delta x_{i-}} \left[2\kappa_i - \delta_x \kappa_i \Delta x_{i+}\right] \nu_{i-1}^n + 2g_i \end{aligned} \tag{2.19}$$

is found. Again, the constants

$$a_i^R = \frac{1}{\Delta x_i \Delta x_{i-}} \left[2\kappa_i - \delta_x \kappa_i \Delta x_{i+}\right] \tag{2.20}$$

$$b_i^R = \frac{2\rho_i c_i}{\Delta t} + \frac{\delta_x \kappa_i [\Delta x_{i+} - \Delta x_{i-}] - 2\kappa_i}{\Delta x_{i-} \Delta x_{i+}} \tag{2.21}$$

$$c_i^R = \frac{1}{\Delta x_i \Delta x_{i+}} \left[2\kappa_i + \delta_x \kappa_i \Delta x_{i-}\right] \tag{2.22}$$

are defined so that the RHS can be simplified to

$$RHS_i = a_i^R \nu_{i-1}^n + b_i^R \nu_i^n + c_i^R \nu_{i+1}^n + 2g_i. \tag{2.23}$$

Combining Equations 2.16 and 2.23 the simplified set of equations appears as

$$a_i^L \nu_{i-1}^{n+1} + b_i^L \nu_i^{n+1} + c_i^L \nu_{i+1}^{n+1} = a_i^R \nu_{i-1}^n + b_i^R \nu_i^n + c_i^R \nu_{i+1}^n + 2g_i. \tag{2.24}$$

2.3 Solution Matrix Representation

The RHS of equation 2.24 can be found explicitly everywhere and is a constant to be calculated. For this derivation when a boundary is encountered, the edge point will be held at zero. This is the simplest boundary condition and will hold the edge values to zero. In the future, these edge points will need to be modified for other boundary conditions. The LHS of equation 2.16 must be solved implicitly using the Thomas algorithm [9]. Below in equation 2.25 is the matrix representation of the tridiagonal system that must be solved:

$$\begin{bmatrix} b_0^L & c_0^L & 0 & & \cdots & & 0 \\ a_1^L & b_1^L & c_1^L & & & & \\ 0 & a_2^L & b_2^L & c_2^L & \ddots & & \vdots \\ & & a_3^L & b_3^L & c_3^L & & \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ & & & & a_{M-1}^L & b_{M-1}^L & c_{M-1}^L \\ 0 & & \cdots & & 0 & a_M^L & b_M^L \end{bmatrix} \begin{bmatrix} \nu_0^{n+1} \\ \nu_1^{n+1} \\ \nu_2^{n+1} \\ \nu_3^{n+1} \\ \vdots \\ \nu_{M-1}^{n+1} \\ \nu_M^{n+1} \end{bmatrix} = \begin{bmatrix} RHS_0 \\ RHS_1 \\ RHS_2 \\ RHS_3 \\ \vdots \\ RHS_{M-1} \\ RHS_M \end{bmatrix} \tag{2.25}$$

In equation 2.25, it is assumed that $\nu_{-1} = 0$ and $\nu_{M+1} = 0$. This is the simplest boundary conditions corresponding to a sink. Other boundary condition are possible by modifying the terms b_0, c_0, a_M, b_M, RHS_0, and RHS_M.

2.4 One-Dimensional Boundary Conditions

2.4.1 Constant Temperature Boundary Condition

Also known as the sink boundary condition, this is the simplest of the boundary conditions. This boundary condition holds a boundary temperature change equal to zero. In this model, the constant temperature boundary condition implies that

$$\nu\Big|_{x_{boundary}} = 0. \tag{2.26}$$

where $x_{boundary}$ can be x_{min} or x_{max}. In the BTEC 1-D Thermal Model, this boundary condition is implemented as an option for use on both surface boundaries.

2.4.2 Insulating Boundary Condition

The insulating boundary condition implies that no energy flows through a boundary. This is translated mathematically to imply that the temperature gradient is equal to zero at the boundary:

$$\frac{\partial \nu}{\partial x}\bigg|_{x_{boundary}} = 0 \tag{2.27}$$

2.4.3 Convective Boundary Condition

The convective boundary condition is a linear boundary condition for which the rate of energy loss on the boundary is proportional to the temperature difference from the ambient temperature. The proportionality constant has a value in power per area per degree and is known as the convective heat transfer rate, h_e:

$$\kappa\frac{\partial \nu}{\partial x}\bigg|_{x_{min}} = h_e(\nu - \nu_\infty) \tag{2.28}$$

$$\kappa\frac{\partial \nu}{\partial x}\bigg|_{x_{max}} = -h_e(\nu - \nu_\infty) \tag{2.29}$$

Here ν is the temperature on the surface and ν_∞ is the external ambient temperature.

2.4.4 Radiative Boundary Condition

A radiative boundary condition follows a Stephan-Boltzmann law where the energy loss rate per unit area is proportional to the difference between the surface and ambient temperatures to the fourth power:

$$\kappa\frac{\partial \nu}{\partial x}\bigg|_{x_{min}} = \sigma\epsilon(T^4 - T_r^4) \tag{2.30}$$

T is in units of Kelvin and T_r is the effective ambient temperature. The effective ambient temperature is a value which is slightly larger than the ambient temperature and corrects for heated air from the surface [5].

This boundary condition is a non-linear boundary condition which cannot easily be represented in a finite difference implementation. The following Taylor approximation is used to linearize the boundary condition [6]:

$$T^4 - T_r^4 \approx 4\,T_r^3(\nu - \nu_r) \tag{2.31}$$

Also, the heat transfer coefficient can be defined as

$$h_r \equiv 4\epsilon\sigma T_r^3. \tag{2.32}$$

The linearized radiation boundary condition equation can be written as

$$\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{min}} = h_r(\nu - \nu_r) \tag{2.33}$$

and

$$\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{max}} = -h_r(\nu - \nu_r). \tag{2.34}$$

2.4.5 Evaporative Boundary Condition

The Lewis analogy for evaporative loss is implemented in the model as an optional surface boundary condition. The method has been employed to include the energy loss from surface evaporation as a function of ambient temperature and relative humidity:

$$\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{max}} = -Q_{vap} \tag{2.35}$$

$$\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{min}} = Q_{vap} \tag{2.36}$$

2.4.6 Combined Boundary Conditions

As these surfaces have differing signs for outward-facing unit normal vectors, energy flows will be described with opposite signs:

$$\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{min}} = h_e(\nu - \nu_\infty) + h_r(\nu - \nu_r) + Q_{vap} \tag{2.37}$$

$$-\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{max}} = h_e(\nu - \nu_\infty) + h_r(\nu - \nu_r) + Q_{vap} \tag{2.38}$$

In addition, values of emissivity and convective heat transfer rates (h_e and h_e) may have differing values, if material properties differ at the two interfaces.

2.4.7 Boundary Condition Derivations

In order to apply the boundary conditions to the Crank-Nicholson Method, the derivatives in Equations 2.37 and 2.38 need to be approximated with a finite difference and then solved for the fictitious points ν_{-1} and ν_M. This needs to be done for ν^n and ν^{n+1}. Lastly, to be able to switch any of the three boundary conditions on or off, three variables a, b, and c will be added:

$$\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{min}} = ah_e(\nu - \nu_\infty) + bh_r(\nu - \nu_r) + cQ_{vap} \tag{2.39}$$

$$-\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{max}} = ah_e(\nu - \nu_\infty) + bh_r(\nu - \nu_r) + cQ_{vap} \tag{2.40}$$

To find ν^n_{-1} the following formulation is used:

$$\kappa_0 \frac{\nu^n_1 - \nu^n_{-1}}{x_1 - x_{-1}} = ah_e(\nu^n_0 - \nu_\infty) + bh_r(\nu^n_0 - \nu_r) + cQ_{vap}(\nu^n_0) \tag{2.41}$$

Let

$$\beta_i \equiv \frac{\kappa_i}{\Delta x_i} \tag{2.42}$$

$$\nu_{-1}^n = \nu_1^n - \frac{1}{\beta_0}[ah_e(\nu_0^n - \nu_\infty) + bh_r(\nu_0^n - \nu_r) + cQ_{vap}(\nu_0^n)] \tag{2.43}$$

and then rearrange terms to obtain

$$\nu_{-1}^n = \nu_1^n + \frac{1}{\beta_0}(ah_e\nu_\infty + bh_r\nu_r) - \frac{1}{\beta_0}(ah_e + bh_r)\nu_0^n - \frac{c}{\beta_0}Q_{vap}(\nu_0^n). \tag{2.44}$$

Finally let

$$q_r = ah_e\nu_\infty + bh_r\nu_r, \tag{2.45}$$

$$q_2 = ah_e + bh_r, \tag{2.46}$$

and

$$\nu_{-1}^n = \nu_1^n + \frac{q_r}{\beta_0} - \frac{q_2}{\beta_0}\nu_0^n - \frac{c}{\beta_0}Q_{vap}(\nu_0^n). \tag{2.47}$$

For ν_M^n, the result is very similar:

$$\nu_M^n = \nu_{M-2}^n + \frac{q_r}{\beta_{M-1}} - \frac{q_2}{\beta_{M-1}}\nu_{M-1}^n - \frac{c}{\beta_{M-1}}Q_{vap}(\nu_{M-1}^n) \tag{2.48}$$

To find ν_{-1}^{n+1} use

$$\kappa_0\frac{\nu_1^{n+1} - \nu_{-1}^{n+1}}{x_1 - x_{-1}} = ah_e(\nu_0^{n+1} - \nu_\infty) + bh_r(\nu_0^{n+1} - \nu_r) + cQ_{vap}(\nu_0^{n+1}). \tag{2.49}$$

In order to find ν_{-1}^{n+1} and ν_M^{n+1}, the value of $Q_{vap}(\nu^{n+1})$ must be approximated using the Taylor series:

$$Q_{vap}(\nu^{n+1}) \approx Q_{vap}(\nu^n) + \left.\frac{dQ_{vap}}{d\nu}\right|_{\nu^n}(\nu^{n+1} - \nu^n) \tag{2.50}$$

For more simplification, let

$$\gamma = c\left.\frac{dQ_{vap}}{d\nu}\right|_{\nu^n}. \tag{2.51}$$

By putting in the approximation and rearranging terms,

$$\nu_{-1}^{n+1} = \nu_1^{n+1} - \frac{1}{\beta_0}[ah_e(\nu_0^{n+1} - \nu_\infty) + bh_r(\nu_0^{n+1} - \nu_r) + cQ_{vap}(\nu_0^n) + \gamma(\nu_0^{n+1} - \nu_0^n)] \tag{2.52}$$

can be found. Next use q_r and q_2 to once more simplify the answer:

$$\nu_{-1}^{n+1} = \nu_1^{n+1} + \frac{\gamma\nu_0^n}{\beta_0} + \frac{q_r}{\beta_0} - \frac{q_2 + \gamma}{\beta_0}\nu_0^{n+1} - \frac{c}{\beta_0}Q_{vap}(\nu_0^n) \tag{2.53}$$

Again, the ν_M^{n+1} point is very similar:

$$\nu_M^{n+1} = \nu_{M-2}^{n+1} + \frac{\gamma\nu_{M-1}^n}{\beta_{M-1}} + \frac{q_r}{\beta_{M-1}} - \frac{q_2 + \gamma}{\beta_{M-1}}\nu_{M-1}^{n+1} - \frac{c}{\beta_{M-1}}Q_{vap}(\nu_{M-1}^n) \tag{2.54}$$

2.5 Boundary Condition Implementation

With all the fictitious end points found, they can be placed into the Crank-Nicholson Method.

Start with equation 2.16 at $i = 0$:

$$LHS_0 = a_0^L v_{-1}^{n+1} + b_0^L v_0^{n+1} + c_0^L v_1^{n+1} \tag{2.55}$$

Incorporating equation 2.53 into equation 2.55 yields

$$\begin{aligned} LHS_0 &= \left(b_0^L - \frac{q_2+\gamma}{\beta_0} a_0^L\right) v_0^{n+1} + (c_0^L + a_0^L) v_1^{n+1} \\ &+ \frac{a_0^L}{\beta_0}\left(\gamma v_0^n + q_r - cQ_{vap}(v_0^n)\right). \end{aligned} \tag{2.56}$$

Note that the last term in equation 2.56 needs to be moved to the RHS because it does not have a v^{n+1} term. This results in the final LHS in being

$$LHS_0 = \left(b_0^L - \frac{q_2+\gamma}{\beta_0} a_0^L\right) v_0^{n+1} + (c_0^L + a_0^L) v_1^{n+1}. \tag{2.57}$$

Start with equation 2.23 at $i = 0$:

$$RHS_0 = a_0^R v_{-1}^n + b_0^R v_0^n + c_0^R v_1^n + 2g_0 \tag{2.58}$$

Putting equation 2.47 into equation 2.58 yields

$$\begin{aligned} RHS_0 &= \left(b_0^R - \frac{q_2}{\beta_0} a_0^R\right) v_0^n + (c_0^R + a_0^R) v_1^n \\ &+ \frac{a_0^R}{\beta_0}\left(q_r - cQ_{vap}(v_0^n)\right) 2g_0. \end{aligned} \tag{2.59}$$

The piece from the LHS is added on. Then

$$\begin{aligned} RHS_0 &= \left(b_0^R - \frac{q_2}{\beta_0} a_0^R\right) v_0^n + (c_0^R + a_0^R) v_1^n \\ &+ \frac{a_0^R}{\beta_0}\left(q_r - cQ_{vap}(v_0^n)\right) - \frac{a_0^L}{\beta_0}\left(\gamma v_0^n + q_r - cQ_{vap}(v_0^n)\right) + 2g_0 \end{aligned} \tag{2.60}$$

is found. For $M-1$, the LHS is

$$LHS_{M-1} = (c_{M-1}^L + a_{M-1}^L) v_{M-2}^{n+1} + \left(b_{M-1}^L - \frac{q_2+\gamma}{\beta_{M-1}} c_{M-1}^L\right) v_{M-1}^{n+1} \tag{2.61}$$

and the RHS is

$$\begin{aligned} RHS_{M-1} &= (c_{M-1}^R + a_{M-1}^R) v_{M-2}^n + \left(b_{M-1}^R - \frac{q_2}{\beta_{M-1}} a_{M-1}^R\right) v_{M-1}^n \\ &+ \frac{c_{M-1}^R}{\beta_{M-1}}\left(q_r - cQ_{vap}(v_{M-1}^n)\right) - \frac{c_{M-1}^L}{\beta_{M-1}}\left(\gamma v_{M-1}^n + q_r - cQ_{vap}(v_{M-1}^n)\right) + 2g_0. \end{aligned} \tag{2.62}$$

Chapter 3

Two-Dimensional Heat Equation

3.1 Heat Equation

Beginning with the two-dimensional heat equation in cylindrical coordinates,

$$\rho c \frac{\partial \nu}{\partial t} = \frac{\kappa}{r}\frac{\partial \nu}{\partial r} + \frac{\partial}{\partial r}\left(\kappa \frac{\partial \nu}{\partial r}\right) + \frac{\partial}{\partial z}\left(\kappa \frac{\partial \nu}{\partial z}\right) + A \tag{3.1}$$

where $\kappa = \kappa(z)$ is the thermal conductivity of the tissue, measured in $\left[\frac{J}{cm\ s\ C}\right]$, $c = c(z)$ is the specific heat of the tissue measured in $\left[\frac{J}{gC}\right]$, $\rho = \rho(z)$ is the density of the tissue measured in $\left[\frac{g}{cm^3}\right]$, $\nu = \nu(z, r, t)$ is the temperature rise at the coordinate (z, r, t) [°C], and $A = A(z, r, t)$ is the source term in units of $\left[\frac{J}{cm^3 s}\right]$.

The numerical approach used in the BTEC is that of Peaceman Rachford [8] and utilizes a split time step to alternate between radial and axial derivatives.

3.2 Finite Difference Representation

Simple numerical central-difference derivative approximations are shown below for both first and second order derivatives. They are shown first for a uniform grid and then for a stretched grid.

3.2.1 Uniform Grid

First Partial Derivative

$$\frac{\partial \nu}{\partial r} = \frac{\nu_{j+1} - \nu_{j-1}}{2\Delta r} \tag{3.2}$$

Second Partial Derivative

$$\frac{\partial^2 \nu}{\partial r^2} = \frac{\nu_{j+1} - 2\nu_j + \nu_{j-1}}{\Delta r^2} \tag{3.3}$$

3.3 Non-Uniform Grid

In order to extend physical space out in the radial direction without using vast computational space or making the grid spaces about the axis too large (where the highest concentration of points are needed in order to model a beam accurately), a stretched grid is required. The following finite differences are geared to handle the variable step sizes in **r**. The notation used is defined below:

$$\Delta r = r_{j+1} - r_{j-1}$$

$$\Delta r_+ = r_{j+1} - r_j$$

$$\Delta r_- = r_j - r_{j-1}$$

3.3.1 First Partial Derivatives

$$\frac{\partial \nu}{\partial r} = \frac{\nu_{j+1} - \nu_{j-1}}{\Delta r} \tag{3.4}$$

3.3.2 Second Partial Derivatives

$$\frac{\partial^2 \nu}{\partial r^2} = \left[\frac{\nu_{j+1} - \nu_j}{\Delta r_+} - \frac{\nu_j - \nu_{j-1}}{\Delta r_-} \right] \frac{2}{\Delta r} \tag{3.5}$$

3.3.3 Simplified Notation

The finite difference approximation for the numerical solution of the heat equation requires the following terms (from 11.13): $\nu(z, r, t) \rightarrow \nu_{i,j}^n$

r $\rightarrow r_j$ j=0,1,2,...,N

z $\rightarrow z_i$ i=0,1,2,...,M

A(z,r,t) $\rightarrow A_{i,j}^n$

Where i is the axial coordinate (z) index, j is the radial coordinate (r) index, and n is the time index representing the time step Δt_n from an initial condition.

The derivation will be simplified further by using the following defined terms:

$\beta = \beta_i^n = \frac{2\rho_i c_i}{\Delta t_n}$

$n' = n + \frac{1}{2}$ $\qquad$ $n'' = n + 1$

$\Delta \kappa_i = \kappa_{i+1} - \kappa_{i-1}$

$\Delta r_j = r_{j+1} - r_{j-1}$ $\qquad$ $\Delta z_i = z_{i+1} - r_{i-1}$

$\Delta r_+ = r_{j+1} - r_j$ $\qquad$ $\Delta r_- = r_j - r_{j-1}$

$\Delta z_+ = z_{i+1} - z_i$ $\qquad$ $\Delta z_- = z_i - z_{i-1}$

$t_n = n\Delta t_n$ (assuming equally-spaced time steps)

3.3.4 Finite Differences

The terms required to represent the two-dimensional heat equation in a finite difference form become

$$\frac{\kappa}{r}\left(\frac{\partial \nu}{\partial r}\right)\Big|_{i,j}^{n} = \frac{\kappa_i}{r_j \Delta r_j}\left(\nu_{i,j+1}^{n} - \nu_{i,j-1}^{n}\right) \tag{3.6}$$

$$\frac{\partial}{\partial r}\left(\kappa\frac{\partial \nu}{\partial r}\right)\Big|_{i,j}^{n} = \frac{2\kappa_i}{\Delta r_j}\left(\frac{\nu_{i,j+1}^{n} - \nu_{i,j}^{n}}{\Delta r_+} - \frac{\nu_{i,j}^{n} - \nu_{i,j-1}^{n}}{\Delta r_-}\right) \tag{3.7}$$

$$\begin{aligned}\frac{\partial}{\partial z}\left(\kappa\frac{\partial \nu}{\partial z}\right)\Big|_{i,j}^{n} &= \frac{2\kappa_i}{\Delta z_i}\left(\frac{\nu_{i+1,j}^{n} - \nu_{i,j}^{n}}{\Delta z_+} - \frac{\nu_{i,j}^{n} - \nu_{i-1,j}^{n}}{\Delta z_-}\right) \\ &\quad + \frac{\delta\kappa}{\Delta z_i^2}\left(\nu_{i+1,j}^{n} - \nu_{i-1,j}^{n}\right)\end{aligned} \tag{3.8}$$

$$\rho c\,\frac{\partial \nu}{\partial t}\Big|_{i,j}^{n'} = \beta_i\left(\nu_{i,j}^{n'} - \nu_{i,j}^{n}\right). \tag{3.9}$$

Axial Boundary Condition

Due to the symmetry of an axial problem

$$\frac{\partial \nu_{i,j}^{n}}{\partial r} = 0\ . \tag{3.10}$$

By evaluation of equation 3.10 using the finite difference representation from equation 3.6, the following relationship can be found across the $r = 0$ boundary:

$$\nu_{i,-1}^{n} = \nu_{i,1}^{n} \tag{3.11}$$

This is convenient because the $\nu_{i,-1}^{n}$ point doesn't exist in the problem space and is just considered a "phantom point."

Also, because of the symmetry of the problem, the phantom grid spaces are evaluated by

$$\begin{aligned}\Delta r_- &= \Delta r_+ \\ \Delta r &= 2\Delta r_+\ .\end{aligned} \tag{3.12}$$

Applying Equations 3.11 and 3.12 from above to equation 3.7, the radial second derivative finite difference approximation at $r = 0$ becomes

$$\frac{\partial}{\partial r}\left(\kappa\frac{\partial \nu}{\partial r}\right)\Big|_{i,0}^{n} = \frac{2\kappa_i}{\Delta r_+^2}\left(\nu_{i,1}^{n} - \nu_{i,0}^{n}\right)\ . \tag{3.13}$$

The relationship in 3.10 does not, however, imply that the equation 3.6 goes to 0 as $\frac{0}{0}$ does not necessarily equate to 0. To find this relationship, L'Hospital's Rule is utilized:

$$\lim_{r\to 0}\frac{1}{r}\frac{\partial \nu}{\partial r} = \frac{\partial^2 \nu}{\partial r^2} \tag{3.14}$$

Combining equation 3.13 and equation 3.14, the first derivative finite difference approximation at $r = 0$ becomes

$$\frac{\kappa}{r}\left(\frac{\partial \nu}{\partial r}\right)\Big|_{i,0}^{n} = \frac{2\kappa_i}{\Delta r_+^2}\left(\nu_{i,1}^{n} - \nu_{i,0}^{n}\right)\ . \tag{3.15}$$

3.3.5 Finite Differences with Simplified Notation

The simplified finite difference expressions are substituted into the heat Equation 3.1. Note that there will be a separate expression for $j = 0$ because of the $r = 0$ boundary condition:

j = 0

$$\begin{aligned}\beta_i^n\left(\nu_{i,0}^{n'}-\nu_{i,0}^n\right) &= \frac{4\kappa_i}{\Delta r_+^2}\left(\nu_{i,1}^n-\nu_{i,0}^n\right)\\ &+ \frac{2\kappa_i}{\Delta z_i}\left(\frac{\nu_{i+1,0}^n-\nu_{i,0}^n}{\Delta z_+}-\frac{\nu_{i,0}^n-\nu_{i-1,0}^n}{\Delta z_-}\right)\\ &+ \frac{\delta\kappa}{\Delta z_i^2}\left(\nu_{i+1,0}^n-\nu_{i-1,0}^n\right)\\ &+ A_{i,0}^{n'}\end{aligned} \tag{3.16}$$

j = 1, 2, ...N − 1

$$\begin{aligned}\beta_i^n\left(\nu_{i,j}^{n'}-\nu_{i,j}^n\right) &= \frac{\kappa_i}{r_j\Delta r_j}\left(\nu_{i,j+1}^n-\nu_{i,j-1}^n\right)\\ &+ \frac{2\kappa_i}{\Delta r_j}\left(\frac{\nu_{i,j+1}^n-\nu_{i,j}^n}{\Delta r_+}-\frac{\nu_{i,j}^n-\nu_{i,j-1}^n}{\Delta r_-}\right)\\ &+ \frac{2\kappa_i}{\Delta z_i}\left(\frac{\nu_{i+1,j}^n-\nu_{i,j}^n}{\Delta z_+}-\frac{\nu_{i,j}^n-\nu_{i-1,j}^n}{\Delta z_-}\right)\\ &+ \frac{\delta\kappa}{\Delta z_i^2}\left(\nu_{i+1,j}^n-\nu_{i-1,j}^n\right)\\ &+ A_{i,j}^{n'}\end{aligned} \tag{3.17}$$

3.3.6 First Half of Peaceman-Rachford

The Peaceman-Rachford method consists of a split time step. The derivation of the first half of the time step is outlined in this section. It is an advantage to symbolically represent all of the radial differential operators as D_r and the axial operators as D_z so the problem can be represented in a familiar matrix-vector language.

Equations 3.16 and 3.17 are rearranged to yield

j = 0

$$
\begin{aligned}
\beta_i^n \nu_{i,0}^{n'} - \beta_i^n \nu_{i,0}^n &= \frac{4\kappa_i}{\Delta r_+^2}\nu_{i,1}^n - \frac{4\kappa_i}{\Delta r_+^2}\nu_{i,0}^n \\
&+ \frac{2\kappa_i}{\Delta z_i \Delta z_+}\nu_{i+1,0}^{n'} - \frac{2\kappa_i}{\Delta z_i \Delta z_+}\nu_{i,0}^{n'} + \frac{2\kappa_i}{\Delta z_i \Delta z_-}\nu_{i+1,0}^{n'} - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\nu_{i,0}^{n'} \\
&+ \frac{\delta\kappa}{\Delta z_i^2}\nu_{i+1,0}^{n'} - \frac{\delta\kappa}{\Delta z_i^2}\nu_{i-1,0}^{n'} \\
&+ A_{i,0}^{n'}
\end{aligned}
$$

$$
\begin{aligned}
&\beta_i^n \nu_{i,0}^{n'} + \left(-\frac{2\kappa_i}{\Delta z_i \Delta z_-} + \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i-1,0}^{n'} + \left(\frac{2\kappa_i}{\Delta z_i \Delta z_+} + \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right)\nu_{i,0}^{n'} + \left(-\frac{2\kappa_i}{\Delta z_i \Delta z_+} - \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i+1,0}^{n'} \\
&= \beta_i^n \nu_{i,0}^n + \left(\frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,1}^n - \left(\frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,0}^n + A_{i,0}^{n'}
\end{aligned}
$$

$$
\begin{aligned}
&\left(+\frac{\delta\kappa}{\Delta z_i^2} - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right)\nu_{i-1,0}^{n'} + \left(\beta_i^n \nu_{i,0}^{n'} + \frac{2\kappa_i}{\Delta z_i}\left(\frac{1}{\Delta z_+} + \frac{1}{\Delta z_-}\right)\right)\nu_{i,0}^{n'} + \left(-\frac{\delta\kappa}{\Delta z_i^2} - \frac{2\kappa_i}{\Delta z_i \Delta z_+}\right)\nu_{i+1,0}^{n'} \\
&= \left(\beta_i^n - \frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,0}^n + \left(\frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,1}^n + A_{i,0}^{n'}
\end{aligned}
\tag{3.18}
$$

j = 1, 2, ...N − 1

$$
\begin{aligned}
\beta_i^n \nu_{i,j}^{n'} - \beta_i^n \nu_{i,j}^n &= \frac{\kappa_i}{r_j \Delta r_j}\nu_{i,j+1}^n - \frac{\kappa_i}{r_j \Delta r_j}\nu_{i,j-1}^n \\
&+ \frac{2\kappa_i}{\Delta r_j \Delta r_+}\nu_{i,j+1}^n - \frac{2\kappa_i}{\Delta r_j \Delta r_+}\nu_{i,j}^n - \frac{2\kappa_i}{\Delta r_j \Delta r_-}\nu_{i,j}^n + \frac{2\kappa_i}{\Delta r_j \Delta r_-}\nu_{i,j-1}^n \\
&+ \frac{2\kappa_i}{\Delta z_i \Delta z_+}\nu_{i+1,j}^n - \frac{2\kappa_i}{\Delta z_i \Delta z_+}\nu_{i,j}^n - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\nu_{i,j}^n + \frac{2\kappa_i}{\Delta z_i \Delta z_-}\nu_{i-1,j}^n \\
&+ \frac{\delta\kappa}{\Delta z_i^2}\nu_{i+1,j}^n - \frac{\delta\kappa}{\Delta z_i^2}\nu_{i-1,j}^n \\
&+ A_{i,j}^{n'}
\end{aligned}
$$

$$
\begin{aligned}
&\beta_i^n \nu_{i,j}^{n'} + \left(-\frac{2\kappa_i}{\Delta z_i \Delta z_-} + \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i-1,j}^{n'} + \left(\frac{2\kappa_i}{\Delta z_i \Delta z_+} + \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right)\nu_{i,j}^{n'} + \left(-\frac{2\kappa_i}{\Delta z_i \Delta z_+} - \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i+1,j}^{n'} \\
&= \beta_i^n \nu_{i,j}^n + \left(\frac{\kappa_i}{r_j \Delta r_j} + \frac{2\kappa_i}{\Delta r_j \Delta r_-}\right)\nu_{i,j-1}^n + \left(-\frac{2\kappa_i}{\Delta r_j \Delta r_+} - \frac{2\kappa_i}{\Delta r_j \Delta r_-}\right)\nu_{i,j}^n + \left(-\frac{\kappa_i}{r_j \Delta r_j} + \frac{2\kappa_i}{\Delta r_j \Delta r_+}\right)\nu_{i,j+1}^n + A_{i,j}^{n'}
\end{aligned}
$$

$$
\begin{aligned}
&\left(\frac{\delta\kappa}{\Delta z_i^2} - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right)\nu_{i-1,j}^{n'} + \left(\beta_i^n + \frac{2\kappa_i}{\Delta z_i}\left(\frac{1}{\Delta z_+} + \frac{1}{\Delta z_-}\right)\right)\nu_{i,j}^{n'} + \left(-\frac{\delta\kappa}{\Delta z_i^2} - \frac{2\kappa_i}{\Delta z_i \Delta z_+}\right)\nu_{i+1,j}^{n'} \\
&= \left(\frac{\kappa_i}{r_j \Delta r_j} + \frac{2\kappa_i}{\Delta r_j \Delta r_-}\right)\nu_{i,j-1}^n + \left(\beta_i^n - \frac{2\kappa_i}{\Delta r_j}\left(\frac{1}{\Delta r_+} + \frac{1}{\Delta r_-}\right)\right)\nu_{i,j}^n + \left(\frac{2\kappa_i}{\Delta r_j \Delta r_+} - \frac{\kappa_i}{r_j \Delta r_j}\right)\nu_{i,j+1}^n + A_{i,j}^{n'}
\end{aligned}
\tag{3.19}
$$

In order to simplify the matrix representation, the following terms are defined:

$$\begin{array}{lll}
\delta z_i^1 \equiv \left(\frac{\delta\kappa}{\Delta z_i^2} - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right) & \delta z_i^2 \equiv \left(\beta_i^n + \frac{2\kappa_i}{\Delta z_i}\left(\frac{1}{\Delta z_+} + \frac{1}{\Delta z_-}\right)\right) & \delta z_i^3 \equiv \left(-\frac{\delta\kappa}{\Delta z_i^2} - \frac{2\kappa_i}{\Delta z_i \Delta z_+}\right) \\
\delta r_{i,0}^1 \equiv 0 & \delta r_{i,0}^2 \equiv \left(\beta_i^n - \frac{4\kappa_i}{\Delta r_+^2}\right) & \delta r_{i,0}^3 \equiv \left(\frac{4\kappa_i}{\Delta r_+^2}\right) \\
\delta r_{i,j}^1 \equiv \left(\frac{\kappa_i}{r_j \Delta r_j} + \frac{2\kappa_i}{\Delta r_j \Delta r_-}\right) & \delta r_{i,j}^2 \equiv \left(\beta_i^n - \frac{2\kappa_i}{\Delta r_j}\left(\frac{1}{\Delta r_+} + \frac{1}{\Delta r_-}\right)\right) & \delta r_{i,j}^3 \equiv \left(\frac{2\kappa_i}{\Delta r_j \Delta r_+} - \frac{\kappa_i}{r_j \Delta r_j}\right)
\end{array}$$

Substitution of these simplification variables into 3.18 and 3.19 yields

$$\delta z_i^1 \nu_{i-1,0}^{n'} + \delta z_i^2 \nu_{i,0}^{n'} + \delta z_i^3 \nu_{i+1,0}^{n'} = \delta r_{i,0}^2 \nu_{i,0}^n + \delta r_{i,1}^3 \nu_{i,0}^n + A_{i,0}^{n'} \tag{3.20}$$

$$\delta z_i^1 \nu_{i-1,j}^{n'} + \delta z_i^2 \nu_{i,j}^{n'} + \delta z_i^3 \nu_{i+1,j}^{n'} = \delta r_{i,j}^1 \nu_{i,j-1}^n + \delta r_{i,j}^2 \nu_{i,j}^n + \delta r_{i,j}^3 \nu_{i,j+1}^n + A_{i,j}^{n'} \, . \tag{3.21}$$

In matrix-vector notation, the first P.R. step becomes

$$\hat{D}_z \nu^{n'} = \hat{D}_r \nu^n + A^{n'} \tag{3.22}$$

$$\omega_m = \hat{D}_r \nu^n + A^{n'}$$

$$\begin{bmatrix} \omega_{i,0} \\ \omega_{i,1} \\ \omega_{i,2} \\ \vdots \\ \vdots \\ \omega_{i,N-2} \\ \omega_{i,N-1} \end{bmatrix} = \begin{bmatrix} \delta r_{i,0}^2 & \delta r_{i,0}^3 & 0 & & \cdots & & 0 \\ \delta r_{i,1}^1 & \delta r_{i,1}^2 & \delta r_{i,1}^3 & & & & \\ 0 & \delta r_{i,2}^1 & \delta r_{i,2}^2 & \delta r_{i,2}^3 & & & \vdots \\ & & \ddots & \ddots & \ddots & & \\ \vdots & & & \ddots & \ddots & \ddots & 0 \\ & & & & \delta r_{i,N-2}^1 & \delta r_{i,N-2}^2 & \delta r_{i,N-2}^3 \\ 0 & & \cdots & & 0 & \delta r_{i,N-1}^1 & \delta r_{i,N-1}^2 \end{bmatrix} \begin{bmatrix} \nu_{i,0} \\ \nu_{i,1} \\ \nu_{i,2} \\ \vdots \\ \vdots \\ \nu_{i,N-2} \\ \nu_{i,N-1} \end{bmatrix} + \begin{bmatrix} A_{i,0} \\ A_{i,1} \\ A_{i,2} \\ \vdots \\ \vdots \\ A_{i,N-2} \\ A_{i,N-1} \end{bmatrix}$$

$$\hat{D}_z \nu^{n'} = \omega_m$$

$$\begin{bmatrix} \delta z_1^2 & \delta z_1^3 & 0 & & \cdots & & 0 \\ \delta z_2^1 & \delta z_2^2 & \delta z_2^3 & & & & \\ 0 & \delta z_3^1 & \delta z_3^2 & \delta z_3^3 & & & \vdots \\ & & \ddots & \ddots & \ddots & & \\ \vdots & & & \ddots & \ddots & \ddots & 0 \\ & & & & \delta z_{N-2}^1 & \delta z_{N-2}^2 & \delta z_{N-2}^3 \\ 0 & & \cdots & & 0 & \delta z_{M-1}^1 & \delta z_{M-1}^2 \end{bmatrix} \begin{bmatrix} \nu_{1,j} \\ \nu_{2,j} \\ \nu_{2,j} \\ \vdots \\ \vdots \\ \nu_{M-2,j} \\ \nu_{M-1,j} \end{bmatrix} = \begin{bmatrix} \omega_{1,j} \\ \omega_{2,j} \\ \omega_{3,j} \\ \vdots \\ \vdots \\ \omega_{M-2,j} \\ \omega_{M-1,j} \end{bmatrix} .$$

3.3.7 Second Half of Peaceman-Rachford

Now equations 3.16 and 3.17 can be arranged into the form of the second half of the Peaceman-Rachford step. This is done in the same manner as the first half of the P.R. step.

j = 0

$$
\begin{aligned}
\beta_i^n \nu_{i,0}^{n''} - \beta_i^n \nu_{i,0}^{n'} &= \frac{4\kappa_i}{\Delta r_+^2}\nu_{i,1}^{n''} - \frac{4\kappa_i}{\Delta r_+^2}\nu_{i,0}^{n''} \\
&+ \frac{2\kappa_i \nu_{i+1,0}^{n'}}{\Delta z_i \Delta z_+} - \frac{2\kappa_i \nu_{i,0}^{n'}}{\Delta z_i \Delta z_+} - \frac{2\kappa_i \nu_{i,0}^{n'}}{\Delta z_i \Delta z_-} + \frac{2\kappa_i \nu_{i-1,0}^{n'}}{\Delta z_i \Delta z_-} \\
&+ \frac{\delta\kappa \nu_{i+1,0}^{n'}}{\Delta z_i^2} - \frac{\delta\kappa \nu_{i-1,0}^{n'}}{\Delta z_i^2} \\
&+ A_{i,0}^{n'}
\end{aligned}
$$

$$
\begin{aligned}
&\beta_i^n \nu_{i,0}^{n''} + \left(\frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,0}^{n''} + \left(-\frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,1}^{n''} = \\
&\beta_i^n \nu_{i,0}^{n'} + \left(\frac{2\kappa_i}{\Delta z_i \Delta z_-} - \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i-1,0}^{n'} + \left(-\frac{2\kappa_i}{\Delta z_i \Delta z_+} - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right)\nu_{i,0}^{n'} + \left(\frac{2\kappa_i}{\Delta z_i \Delta z_+} + \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i+1,0}^{n'} + A_{i,0}^{n'}
\end{aligned}
$$

$$
\begin{aligned}
&\left(\beta_i^n + \frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,0}^{n''} + \left(-\frac{4\kappa_i}{\Delta r_+^2}\right)\nu_{i,1}^{n''} = \\
&\left(-\frac{\delta\kappa}{\Delta z_i^2} + \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right)\nu_{i-1,0}^{n'} + \left(\beta_i^n \nu_{i,0}^{n'} - \frac{2\kappa_i}{\Delta z_i}\left(\frac{1}{\Delta z_+} + \frac{1}{\Delta z_-}\right)\right)\nu_{i,0}^{n'} + \left(\frac{\delta\kappa}{\Delta z_i^2} + \frac{2\kappa_i}{\Delta z_i \Delta z_+}\right)\nu_{i+1,0}^{n'} + A_{i,0}^{n'}
\end{aligned}
\tag{3.23}
$$

j = 1, 2, ...N − 1

$$
\begin{aligned}
\beta_i^n \nu_{i,j}^{n''} - \beta_i^n \nu_{i,j}^{n'} &= \frac{\kappa_i}{r_j \Delta r_j}\nu_{i,j+1}^{n'} - \frac{\kappa_i}{r_j \Delta r_j}\nu_{i,j-1}^{n'} \\
&+ \frac{2\kappa_i}{\Delta r_j \Delta r_+}\nu_{i,j+1}^{n'} - \frac{2\kappa_i}{\Delta r_j \Delta r_+}\nu_{i,j}^{n'} - \frac{2\kappa_i}{\Delta r_j \Delta r_-}\nu_{i,j}^{n'} + \frac{2\kappa_i}{\Delta r_j \Delta r_-}\nu_{i,j-1}^{n'} \\
&+ \frac{2\kappa_i}{\Delta z_i \Delta z_+}\nu_{i+1,j}^{n'} - \frac{2\kappa_i}{\Delta z_i \Delta z_+}\nu_{i,j}^{n'} - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\nu_{i,j}^{n'} + \frac{2\kappa_i}{\Delta z_i \Delta z_-}\nu_{i-1,j}^{n'} \\
&+ \frac{\delta\kappa}{\Delta z_i^2}\nu_{i+1,j}^{n'} - \frac{\delta\kappa}{\Delta z_i^2}\nu_{i-1,j}^{n'} \\
&+ A_{i,j}^{n'}
\end{aligned}
$$

$$
\begin{aligned}
&\beta_i^n \nu_{i,j}^{n''} + \left(\frac{\kappa_i}{r_j \Delta r_j} - \frac{2\kappa_i}{\Delta r_j \Delta r_-}\right)\nu_{i,j-1}^{n''} + \left(\frac{2\kappa_i}{\Delta r_j \Delta r_+} + \frac{2\kappa_i}{\Delta r_j \Delta r_-}\right)\nu_{i,j}^{n''} + \left(-\frac{\kappa_i}{r_j \Delta r_j} - \frac{2\kappa_i}{\Delta r_j \Delta r_+}\right)\nu_{i,j+1}^{n''} \\
&= \beta_i^n \nu_{i,j}^{n'} + \left(\frac{2\kappa_i}{\Delta z_i \Delta z_-} - \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i-1,j}^{n'} + \left(-\frac{2\kappa_i}{\Delta z_i \Delta z_+} - \frac{2\kappa_i}{\Delta z_i \Delta z_-}\right)\nu_{i,j}^{n'} + \left(\frac{2\kappa_i}{\Delta z_i \Delta z_+} + \frac{\delta\kappa}{\Delta z_i^2}\right)\nu_{i+1,j}^{n'} + A_{i,j}^{n'}
\end{aligned}
$$

$$\begin{aligned}&\left(\frac{\kappa_i}{r_j\Delta r_j}-\frac{2\kappa_i}{\Delta r_j\Delta r_-}\right)\nu_{i,j-1}^{n''}+\left(\beta_i^n+\frac{2\kappa_i}{\Delta r_j}\left(\frac{1}{\Delta r_+}+\frac{1}{\Delta r_-}\right)\right)\nu_{i,j}^{n''}+\left(-\frac{2\kappa_i}{\Delta r_j\Delta r_+}-\frac{\kappa_i}{r_j\Delta r_j}\right)\nu_{i,j+1}^{n''}\\&=\left(-\frac{\delta\kappa}{\Delta z_i^2}+\frac{2\kappa_i}{\Delta z_i\Delta z_-}\right)\nu_{i-1,j}^{n'}+\left(\beta_i^n-\frac{2\kappa_i}{\Delta z_i}\left(\frac{1}{\Delta z_+}+\frac{1}{\Delta z_-}\right)\right)\nu_{i,j}^{n'}+\left(\frac{\delta\kappa}{\Delta z_i^2}+\frac{2\kappa_i}{\Delta z_i\Delta z_+}\right)\nu_{i+1,j}^{n'}+A_{i,j}^{n'}\end{aligned}\tag{3.24}$$

Again, in order to simplify the matrix representation, the following terms are defined:

$$\begin{array}{lll}\delta z_i^1\equiv\left(-\frac{\delta\kappa}{\Delta z_i^2}+\frac{2\kappa_i}{\Delta z_i\Delta z_-}\right) & \delta z_i^2\equiv\left(\beta_i^n-\frac{2\kappa_i}{\Delta z_i}\left(\frac{1}{\Delta z_+}+\frac{1}{\Delta z_-}\right)\right) & \delta z_i^3\equiv\left(\frac{\delta\kappa}{\Delta z_i^2}+\frac{2\kappa_i}{\Delta z_i\Delta z_+}\right)\\ \delta r_{i,0}^1\equiv 0 & \delta r_{i,0}^2\equiv\left(\beta_i^n+\frac{4\kappa_i}{\Delta r_+^2}\right) & \delta r_{i,0}^3\equiv\left(-\frac{4\kappa_i}{\Delta r_+^2}\right)\\ \delta r_{i,j}^1\equiv\left(\frac{\kappa_i}{r_j\Delta r_j}-\frac{2\kappa_i}{\Delta r_j\Delta r_-}\right) & \delta r_{i,j}^2\equiv\left(\beta_i^n+\frac{2\kappa_i}{\Delta r_j}\left(\frac{1}{\Delta r_+}+\frac{1}{\Delta r_-}\right)\right) & \delta r_{i,j}^3\equiv\left(-\frac{2\kappa_i}{\Delta r_j\Delta r_+}-\frac{\kappa_i}{r_j\Delta r_j}\right)\end{array}$$

Substitution of these simplification variables into 3.23 and 3.24 yields

$$\delta r_{i,0}^2\nu_{i,0}^{n''}+\delta r_{i,1}^3\nu_{i,0}^{n''}=\delta z_i^1\nu_{i-1,0}^{n'}+\delta z_i^2\nu_{i,0}^{n'}+\delta z_i^3\nu_{i+1,0}^{n'}+A_{i,0}^{n'}\tag{3.25}$$

$$\delta r_{i,j}^1\nu_{i,j-1}^{n''}+\delta r_{i,j}^2\nu_{i,j}^{n''}+\delta r_{i,j}^3\nu_{i,j+1}^{n''}=\delta z_i^1\nu_{i-1,j}^{n'}+\delta z_i^2\nu_{i,j}^{n'}+\delta z_i^3\nu_{i+1,j}^{n'}+A_{i,j}^{n'}\,.\tag{3.26}$$

In matrix-vector notation, the second P.R. step becomes

$$\hat{D}_r\nu^{n''}=\hat{D}_z\nu^{n'}+A^{n'}\tag{3.27}$$

$$\omega=\hat{D}_z\nu^{n'}+A^{n'}$$

$$\begin{bmatrix}\omega_{1,j}\\ \omega_{2,j}\\ \omega_{3,j}\\ \vdots\\ \vdots\\ \omega_{M-2,j}\\ \omega_{M-1,j}\end{bmatrix}=\begin{bmatrix}\delta z_0^2 & \delta z_0^3 & 0 & & \cdots & & 0\\ \delta z_1^1 & \delta z_1^2 & \delta z_1^3 & & & & \\ 0 & \delta z_2^1 & \delta z_2^2 & \delta z_2^3 & & & \vdots\\ & & \ddots & \ddots & \ddots & & \\ \vdots & & & \ddots & \ddots & \ddots & 0\\ & & & & \delta z_{M-2}^1 & \delta z_{M-2}^2 & \delta z_{M-2}^3\\ 0 & & \cdots & & 0 & \delta z_{M-1}^1 & \delta z_{M-1}^2\end{bmatrix}\begin{bmatrix}\nu_{1,j}\\ \nu_{2,j}\\ \nu_{3,j}\\ \vdots\\ \vdots\\ \nu_{M-2,j}\\ \nu_{M-1,j}\end{bmatrix}+\begin{bmatrix}A_{1,j}^{n'}\\ A_{2,j}^{n'}\\ A_{3,j}^{n'}\\ \vdots\\ \vdots\\ A_{M-1,j}^{n'}\\ A_{M-1,j}^{n'}\end{bmatrix}$$

$$\hat{D}_r\nu^{n''}=\omega$$

$$\begin{bmatrix}\delta r_{i,0}^2 & \delta r_{i,0}^3 & 0 & & \cdots & & 0\\ \delta r_{i,1}^1 & \delta r_{i,1}^2 & \delta r_{i,1}^3 & & & & \\ 0 & \delta r_{i,2}^1 & \delta r_{i,2}^2 & \delta r_{i,2}^3 & & & \vdots\\ & & \ddots & \ddots & \ddots & & \\ \vdots & & & \ddots & \ddots & \ddots & 0\\ & & & & \delta r_{i,N-2}^1 & \delta r_{i,N-2}^2 & \delta r_{i,N-2}^3\\ 0 & & \cdots & & 0 & \delta r_{i,N-1}^1 & \delta r_{i,N-1}^2\end{bmatrix}\begin{bmatrix}\nu_{i,0}\\ \nu_{i,1}\\ \nu_{i,2}\\ \vdots\\ \vdots\\ \nu_{i,N-1}\\ \nu_{i,N-1}\end{bmatrix}=\begin{bmatrix}\omega_{i,0}\\ \omega_{i,1}\\ \omega_{i,2}\\ \vdots\\ \vdots\\ \omega_{i,N-1}\\ \omega_{i,N-1}\end{bmatrix}.$$

3.4 Boundary Conditions

3.4.1 Constant-Temperature Boundary Condition

Also known as the sink, this is the simplest of the boundary conditions. This boundary condition holds a boundary temperature change equal to zero. In the model, this implies the following:

$$\nu|_{r_{max}} = 0 \tag{3.28}$$

$$\nu|_{z_{min}} = 0$$

$$\nu|_{z_{max}} = 0 \tag{3.29}$$

In the implementation of the BTEC Thermal Model, this boundary condition has been implemented for optional use on all boundary surfaces.

3.4.2 Insulating Boundary Condition

The insulating boundary condition implies that no energy flows through a boundary. This is represented mathematically as a zero temperature gradient at the boundary:

$$\left.\frac{\partial \nu}{\partial r}\right|_{r_{max}} = 0 \tag{3.30}$$

$$\left.\frac{\partial \nu}{\partial z}\right|_{z_{min}} = 0$$

$$\left.\frac{\partial \nu}{\partial z}\right|_{z_{max}} = 0 \tag{3.31}$$

Applying a central finite difference to equation 3.30, it is found that

$$\nu^n_{i,N} = \nu^n_{i,N-2} \tag{3.32}$$

where r_{max} corresponds to j=N-1.

The first half-time step in the finite difference Peaceman Rachford Method (implicit in z) is

$$\delta z^1_i \nu^{n'}_{i-1,N-1} + \delta z^2_i \nu^{n'}_{i,N-1} + \delta z^3_i \nu^{n'}_{i+1,N-1} = \delta r^1_{i,N-1} \nu^n_{i,N-2} + \delta r^2_{i,N-1} \nu^n_{i,N-1} + \delta r^3_{i,N-1} \nu^n_{i,N} + A^{n'}_{i,N-1} \, . \tag{3.33}$$

Incorporating equation 3.32 into 3.33, the following relationship is derived to be implemented for an insulating boundary condition:

$$\delta z^1_i \nu^{n'}_{i-1,N-1} + \delta z^2_i \nu^{n'}_{i,N-1} + \delta z^3_i \nu^{n'}_{i+1,N-1} = \left(\delta r^1_{i,j} + \delta r^3_{i,j}\right) \nu^n_{i,N-2} + \delta r^2_{i,j} \nu^n_{i,N-1} + A^{n'}_{i,N-1} \tag{3.34}$$

The second half-time step of the Peaceman Rachford Method (implicit in r) has the finite difference representation shown below:

$$\delta r^1_{i,N-1} \nu^{n''}_{i,N-2} + \delta r^2_{i,N-1} \nu^{n''}_{i,N-1} + \delta r^3_{i,N-1} \nu^{n''}_{i,N} = \delta z^1_i \nu^{n'}_{i-1,N-1} + \delta z^2_i \nu^{n'}_{i,N-1} + \delta z^3_i \nu^{n'}_{i+1,N-1} + A^{n'}_{i,N-1} \tag{3.35}$$

Applying equation 3.32 into 3.35, the following relationship is derived to be implemented for an insulating boundary condition:

$$\left(\delta r^1_{i,N-1} + \delta r^3_{i,N-1}\right) \nu^{n''}_{i,N-2} + \delta r^2_{i,N-1} \nu^{n''}_{i,N-1} = \delta z^1_i \nu^{n'}_{i-1,N-1} + \delta z^2_i \nu^{n'}_{i,N-1} + \delta z^3_i \nu^{n'}_{i+1,N-1} + A^{n'}_{i,N-1} \tag{3.36}$$

3.4.3 Convective Boundary Condition

The convective boundary condition is a linear boundary condition for which the rate of energy loss on the boundary is proportional to the temperature difference from the ambient temperature. The proportionality constant has a value in power per area per degree and is known as the convective heat transfer rate, h_e. In the case of the 2-D model, this boundary condition has been implemented as an optional boundary condition for the z_{min} and z_{max} surfaces:

$$\kappa \frac{\partial \nu}{\partial z}\bigg|_{z_{min}} = h_e(\nu - \nu_\infty) \tag{3.37}$$

Here, ν_∞ is the ambient temperature, relative to the baseline tissue temperature.

3.4.4 Radiative Boundary Condition

A radiative boundary condition follows the Stephan-Boltzmann law where the energy loss rate per unit area is proportional to the difference between the absolute surface and ambient temperatures to the fourth power:

$$\kappa \frac{\partial \nu}{\partial x}\bigg|_{x_{min}} = \sigma\epsilon(\theta^4 - \theta_r^4) \tag{3.38}$$

The temperatures, θ and θ_r, are the surface temperature and effective ambient temperature, in Kelvin. The effective ambient temperature is slightly higher than the actual ambient temperature, and corrects for heated air from the surface. This boundary condition is a non-linear boundary condition which cannot easily be represented in a finite difference implementation.

3.4.5 Evaporative Boundary Condition

The Lewis analogy for evaporative loss is implemented in the model as an optional surface boundary condition. The method has been employed to include the energy loss from surface evaporation as a function of ambient temperature and relative humidity:

$$\kappa \frac{\partial \nu}{\partial z}\bigg|_{z_{min}} = Q_{vap} \tag{3.39}$$

3.4.6 Combined Boundary Conditions

In order to implement a meaningful surface boundary condition for the 2-D model in cylindrical coordinates, it is important to place surfaces at the minimum and maximum z coordinates. As these surfaces have differing signs for outward-facing unit normal vectors, energy flows will be described with opposite signs:

$$\kappa \frac{\partial \nu}{\partial z}\bigg|_{z_{min}} = h_e(\nu - \nu_\infty) + \sigma\epsilon(\theta^4 - \theta_r^4) + Q_{vap} \tag{3.40}$$

$$-\kappa \frac{\partial \nu}{\partial z}\bigg|_{z_{max}} = h_e(\nu - \nu_\infty) + \sigma\epsilon(\theta^4 - \theta_r^4) + Q_{vap} \tag{3.41}$$

In addition, values of emissivity and convective heat transfer rates may have differing values if material properties differ at the two interfaces. The implementation at the z_{min} surface within the finite difference method are considered and then the additional parameters required for code implementation are examined.

3.5 Boundary Condition Implementation

In the BTEC (2-D) Peaceman Rachford Method, there is an alternating-direction implicit method. For each half time step, the system of equations implementing boundary conditions is different. The half-step implicit in z is considered first. The following equation utilizes the notation for differential operators.

$$\delta z_i^1 \nu_{i-1,j}^{n'} + \delta z_i^2 \nu_{i,j}^{n'} + \delta z_i^3 \nu_{i+1,j}^{n'} = \delta r_{i,j}^1 \nu_{i,j-1}^{n} + \delta r_{i,j}^2 \nu_{i,j}^{n} + \delta r_{i,j}^3 \nu_{i,j+1}^{n} + A_{i,j}^{n'} \tag{3.42}$$

The z_{min} coordinate is indexed in the model as $i = 0$, such that the surface boundary condition can be represented as

$$\delta z_0^1 \nu_{-1,j}^{n'} + \delta z_0^2 \nu_{0,j}^{n'} + \delta z_0^3 \nu_{1,j}^{n'} = \delta r_{0,j}^1 \nu_{0,j-1}^{n} + \delta r_{0,j}^2 \nu_{0,j}^{n} + \delta r_{0,j}^3 \nu_{0,j+1}^{n} + A_{0,j}^{n'} \, . \tag{3.43}$$

The $\nu_{-1,j}^{n'}$ coordinate is a fictitious point, eliminated by solving for this coordinate in the boundary condition representation:

$$\nu_{-1,j}^{n'} = \nu_{1,j}^{n'} + \frac{1}{\beta}(h_1 + h_r)\nu_{0,j}^{n'} + \frac{q_r}{\beta} + \frac{Q_{vap}(\nu_{0,j}^{n})}{\beta} + \frac{\gamma(\nu_{0,j}^{n})}{\beta}(\nu_{0,j}^{n'} - \nu_{0,j}^{n}) \tag{3.44}$$

The fictitious point is eliminated through substitution to eliminate $\nu_{-1,j}^{n'}$ in the finite difference representation. The LHS of the finite difference representation at the boundary becomes

$$\begin{aligned} LHS \; = \nu_{0,j}^{n'} \left[\delta z_0^1 \left(\frac{1}{\beta}(h_1 + h_r) + \frac{\gamma(\nu_{0,j}^{n})}{\beta} \right) + \delta z_0^2 \right] + \nu_{1,j}^{n'} \left[\delta z_0^1 + \delta z_0^3 \right] \\ + \delta z_0^1 \left[\frac{q_r}{\beta} + \frac{Q_{vap}(\nu_{0,j}^{n})}{\beta} \right] - \delta z_0^1 \left[\frac{\gamma(\nu_{0,j}^{n})}{\beta} \nu_{0,j}^{n} \right] . \end{aligned} \tag{3.45}$$

The second half-time step of the Peaceman Rachford Method (implicit in r) has the finite difference representation shown below:

$$\delta r_{i,j}^1 \nu_{i,j-1}^{n''} + \delta r_{i,j}^2 \nu_{i,j}^{n''} + \delta r_{i,j}^3 \nu_{i,j+1}^{n''} = \delta z_i^1 \nu_{i-1,j}^{n'} + \delta z_i^2 \nu_{i,j}^{n'} + \delta z_i^3 \nu_{i+1,j}^{n'} + A_{i,j}^{n'} \tag{3.46}$$

At the minimum z coordinate boundary, $i = 0$, yielding

$$\delta r_{0,j}^1 \nu_{0,j-1}^{n''} + \delta r_{0,j}^2 \nu_{0,j}^{n''} + \delta r_{0,j}^3 \nu_{0,j+1}^{n''} = \delta z_0^1 \nu_{-1,j}^{n'} + \delta z_0^2 \nu_{0,j}^{n'} + \delta z_0^3 \nu_{1,j}^{n'} + A_{0,j}^{n'} \, , \tag{3.47}$$

and the explicit boundary condition representation is

$$\nu_{-1,j}^{n'} = \nu_{1,j}^{n'} + \frac{1}{\beta}(h_1 + h_r)\nu_{0,j}^{n'} + \frac{q_r}{\beta} + \frac{1}{\beta} Q_{vap}(\nu_{0,j}^{n'}). \tag{3.48}$$

The boundary condition relationship in equation 3.48 is used to replace the phantom point in the finite difference representations. The right hand side of second P.R. step then becomes

$$\begin{aligned} RHS \; = \nu_{0,j}^{n'} \left[\delta z_0^1 \frac{1}{\beta}(h_1 + h_r) + \delta z_0^2 \right] + \nu_{1,j}^{n'} \left[\delta z_0^1 + \delta z_0^3 \right] \\ + \delta z_0^1 \left[\frac{q_r}{\beta} + \frac{1}{\beta} Q_{vap}(\nu_{0,j}^{n}) \right] + A_{0,j}^{n'} \, . \end{aligned} \tag{3.49}$$

The implementation of surface boundary conditions involve several functions determined by absolute temperature in Kelvin, temperature in Celsius, or a relative temperature. The finite difference solution is implemented in terms of change in temperature. The code tracks various temperatures in terms of the current finite difference solution or in terms of measured temperature.

In addition, the software implementation of the surface boundary conditions will allow the user to simply switch a given effect on or off. The code will use the same solution for any combination of convective, radiative, and evaporative effects.

Various temperatures are defined to simplify the notation of the boundary conditions. Temperatures in Kelvin are denoted by, θ, temperatures in Celsius are denoted by, T. The convention of relative temperature has already been used throughout this paper, and are denoted by ν.

$$\nu = T - T_0 \tag{3.50}$$

where T_0 is the baseline tissue temperature. The relative ambient temperature is

$$\nu_\infty = T_\infty - T_0 \ . \tag{3.51}$$

The effective relative ambient temperature is

$$\nu_r = T_r - T_0 \tag{3.52}$$

In addition, the variables a, b, and c are defined to represent switch values for each of the convective, radiative, and evaporative boundary conditions, respectively. The following intermediate values are also used to simplify notation.

$$h_r = 4\epsilon\sigma T_r^3 \tag{3.53}$$

$$q_2 = ah_1 + bh_r \tag{3.54}$$

$$q_r = ah_1\nu_\infty + bh_r\nu_r \tag{3.55}$$

$$\gamma = c\frac{dQ_{vap}}{dT}\Big|_\nu \tag{3.56}$$

The boundary condition at the z_{min} surface is then given by

$$\kappa\frac{\partial\nu}{\partial z}\Big|_{z_{min}} = ah_r(\nu - \nu_\infty) + b\sigma\epsilon(\theta^4 - \theta_r^4) + cQ_{vap} \ , \tag{3.57}$$

where a, b, and c may be assigned differing values on the z_{max} surface.
The derivation of the implicit and explicit (in z) finite difference representation of the $i = 0$, or z_{min} grid positions are obtained in the following equations:

$$\begin{aligned} &\nu_{0,j}^{n'}\left[\delta z_0^2 - \delta z_0^1(\frac{q_2}{\beta} + \frac{\gamma}{\beta})\right] + \nu_{1,j}^{n'}(\delta z_0' + \delta z_0^3) \\ &\qquad + \delta z_0'\left[\frac{q_r}{\beta} - \frac{c}{\beta}Q_{vap}(\nu_{0,j}^n) + \frac{\gamma}{\beta}\nu_{0,j}^n\right] \\ &= \delta r_{0,j}^1\nu_{0,j-1}^n + \delta r_{0,j}^2\nu_{0,j}^n + \delta r_{0,j}^3\nu_{0,j+1}^n + A_{0,j}^{n'} \end{aligned} \tag{3.58}$$

$$\begin{aligned} &\delta r_{0,j}^1\nu_{0,j-1}^{n''} + \delta r_{0,j}^2\nu_{0,j}^{n''} + \delta r_{0,j}^3\nu_{0,j+1}^{n''} \\ &= \nu_{0,j}^{n'}\left[\delta z_0^2 - \delta z_0^1(\frac{q_2}{\beta} + \frac{\gamma}{\beta})\right] + \nu_{1,j}^{n'}(\delta z_0' + \delta z_0^3) \\ &\qquad + \delta z_0'\left[\frac{q_r}{\beta} - \frac{c}{\beta}Q_{vap}(\nu_{0,j}^n)\right] + A_{0,j}^{n'} \end{aligned} \tag{3.59}$$

The same boundary conditions at the z_{max} surface are also defined.

Note: In the BTEC model implementation, the $nz-1$ index is a position on the boundary and the nz index represents a fictional or phantom grid-point, and repeats the derivation for the equations above:

$$\begin{aligned} \nu^{n'}_{M,j}\left[\delta z^2_M - \delta z^3_M(\frac{-q_2}{\beta}+\frac{-\gamma}{\beta})\right] + \nu^{n'}_{M-1,j}(\delta z'_M + \delta z^3_M) \\ +\delta z^3_M\left[\frac{q_r}{\beta} - \frac{c}{\beta}Q_{vap}(\nu^n_{M,j}) + \frac{\gamma}{\beta}\nu^n_{M,j}\right] \\ = \delta r^1_{M,j}\nu^n_{M,j-1} + \delta r^2_{M,j}\nu^n_{M,j} + \delta r^3_{M,j}\nu^n_{M,j+1} + A^{n'}_{M,j} \end{aligned} \tag{3.60}$$

$$\begin{aligned} \delta r^1_{M,j}\nu^{n''}_{M,j-1} + \delta r^2_{M,j}\nu^{n''}_{M,j} + \delta r^3_{M,j}\nu^{n''}_{M,j+1} \\ = \nu^{n'}_{M,j}\left[\delta z^2_M - \delta z^3_M(\frac{q_2}{\beta})\right] + \nu^{n'}_{M-1,j}(\delta z'_M + \delta z^3_M) \\ +\delta z^3_M\left[\frac{q_r}{\beta} - \frac{c}{\beta}Q_{vap}(\nu^n_{M,j})\right] + A^{n'}_{M,j} \end{aligned} \tag{3.61}$$

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 4

Source Terms and Perfusion Effects

4.1 Source Term Representation

The BTEC thermal model has the capability of using several different types of source. Ultimately, a 3-dimensional, cylindrically symmetric, time dependent, source term with dimensions of W/cm^3 is needed to solve the heat equation. All source terms (from multiple lasers, perfusion, etc.) can be summed into one function, denoted by $A(z, r, t)$. This function is then the single source term for the Heat Equation.
BTEC thermal can compute a Beer's Law source based on the profile of the beam and the absorption. With the Beer's Law source the model is set up to compute the source term for a flat-top, Gaussian, and annular profile beams. The BTEC model can also utilize wave propagation. The finite difference wave propagation and Hankel wave propagation methods compute the source term based on the initial electric field distribution and the current refractive index of each node. The refractive index varies with temperature and the source term is recomputed. The wave propagation methods and their derivations are discussed in more detail in Chapter 6.

The heat source term for the formulation of the thermal model was examined through two differing approaches. The different approaches are referred to as the optical model and the radio frequency model. In the optical effects formulation, the time-dependent solution to the Heat Equation is determined for a source term defined by equation 4.5. A localized heat source term estimated from a simple linear absorption is commonly used for optical frequencies. The time dependent amplitude is assumed to be a square-pulse temporal profile with a Gaussian, top-hat, or annular spatial distribution.

4.1.1 Beer's Law

The 3-dimensional, time dependent, source term computed by Beer's Law is

$$A(z, r, t) = \mu_a I_0(r, t) e^{-\mu_a z} \ . \tag{4.1}$$

The exponential decay in the z direction is dependent on the absorption coefficient, μ_a. $I_0(r, t)$ is the incident, spacial beam profile. The three beam profiles mentioned are represented below.

Flat-Top

$$I_0(r, t) = \begin{cases} I_0, & r \leq \sigma \\ 0, & r > \sigma \end{cases} \tag{4.2}$$

Gaussian

$$I_0(r, t) = I_0 e^{-r^2/\sigma^2} \tag{4.3}$$

Annular

$$I_0(r,t) = \begin{cases} I_0, & \sigma_1 \leq r \leq \sigma_2 \\ 0, & r < \sigma_1, r > \sigma_2 \end{cases} \tag{4.4}$$

The beam radius is given by ,*sigma*, and the Peak Incident Irradiance is I_0.

4.1.2 SAR

$$A(z,r,t) = h(z,r)H_0(\lambda,t)\mu_a(z,\lambda) \tag{4.5}$$

This source term provides a time-dependent description of the linear absorption of optical energy as a function of depth in the tissue, complete with spectral and radial dependence of energy being absorbed. The variable λ refers to the wavelength of the THz source. The function $h(z,r)$ specifies the relative irradiance for a given position in the cylindrical coordinate system and includes losses due to linear absorption. This term also addresses the focusing of the beam through the tissues using a specified beam waist location and a hyperbolic function to assign beam radius as a function of position. The function $H_0(\lambda,t)$ provides the maximum irradiance per wavelength division at a given time $[W/cm^2/nm]$. The value of $\mu_a(z,\lambda)$ represents the absorption coefficient $[1/cm]$ at a given wavelength within the tissue, which is determined by the tissue type at the given axial depth, z $[cm]$.

In the radio frequency formulation of the source term, a one-dimensional Finite Difference Time Domain (FDTD) method is used to predict the specific absorption rate (SAR) within single- and multi-layer homogeneous skin slab models. The FDTD method is an explicit time-domain numerical approach for solving Maxwell's equations in a discretized space[14, 11]. This method has become widely used for predicting the electromagnetic fields and energy absorption rates within digital voxelized models. Equations 4.6-4.7 below give the FDTD update equations for the E_x and H_y fields, respectively:

$$E_x|_k^{n+\frac{1}{2}} = \left[\frac{1-\frac{\sigma_k\Delta t}{2\epsilon_k}}{1+\frac{\sigma_k\Delta t}{2\epsilon_k}}\right] E_x|_k^{n-\frac{1}{2}} + \left[\frac{\frac{\Delta t}{\epsilon_k}}{1+\frac{\sigma_k\Delta t}{2\epsilon_k}}\right]\left[\frac{H_y|_{k-\frac{1}{2}}^n - H_y|_{k+\frac{1}{2}}^n}{\Delta z}\right] \tag{4.6}$$

$$H_y|_k^{n+1} = \left[\frac{1-\frac{\sigma_k^*\Delta t}{2\mu_k}}{1+\frac{\sigma_k^*\Delta t}{2\mu_k}}\right] H_x|_k^n + \left[\frac{\frac{\Delta t}{\mu_k}}{1+\frac{\sigma_k^*\Delta t}{2\mu_k}}\right]\left[\frac{E_x|_{k-\frac{1}{2}}^{n+\frac{1}{2}} - E_x|_{k+\frac{1}{2}}^{n+\frac{1}{2}}}{\Delta z}\right] \tag{4.7}$$

In Equations 4.6-4.7, σ is the conductivity in Siemens per meter, μ is the permeability in Henrys per meter, and σ^* is the magnetic loss term in ohms per meter. The magnetic loss is assumed to be zero and the localized SAR values within the tissue are calculated by equation 4.8. The index k represents the discretized axial coordinate, $z = k(\Delta z)$, and n indicates the time step, $t = n(\Delta t)$. The SAR value is subsequently defined from the electric field strength and the density of the tissue, as given by equation 4.8:

$$SAR = \frac{\sigma|E|^2}{\rho} \tag{4.8}$$

In equation 4.8, ρ is the density $[kg/cm^3]$, and E is the root-mean-square of the electric field.

4.2 Perfusion Effects

The BTEC thermal model employs a non-localized, uniform perfusion term to simulate the convective loss of energy due to blood flow. The term physically represents a uniform constant exchange of mass at each tissue location, with the entering mass having a temperature of the reference tissue temperature (body temperature). The form of the perfusion source term equation is given below:

$$\dot{q}(z,r,t) = -w_p(z)\, c(z)\, \nu(z,r,t) \tag{4.9}$$

The perfusion flow-rate term, w_p, expressed in $W/cm^3/s$, is assigned to each tissue layer type within the model and added to the current source term calculated from the emitter's properties. The values c and v represent specific heat and temperature rise in the tissue, respectively. The perfusion term, $c(z)$, currently implemented is constant with temperature. Typical values can be found in the text by Welch[13].

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Gaussian Beam Propagation

Two forms of propagation analysis are currently employed in the BTEC model: a Gaussian Beam Propagation Method (GBP) and wave propagation. The GBP allows for Gaussian beams to be translated through linear optics. It is mostly used to compute the initial conditions for the wave propagators employed in the BTEC project. For the Gaussian Beam Propagator in this project, a general expression using, the ABCD matrix method common in optical transformations is needed to calculate how a laser beam propagates through linear geometric optics. Many books and articles solve this problem with the condition (or assumption) that the lens or optical component is placed at the waist of the approaching beam. The goal is an expression for an optical system placed in any arbitrary position along the approaching beam path.

The electric field E for a Gaussian beam can be defined as

$$E = A\frac{\omega_0}{\omega} exp\left(\left[\frac{-r}{\omega^2}\right] + i\left[k(z - z_\omega) - tan^{-1}\left(\frac{z - z_\omega}{z_R}\right)\right] + i\left[\frac{kr^2}{2R(z)}\right] - i\left[\frac{kr^4}{8R^3(z)}\right]\right) , \tag{5.1}$$

where ω_0 is the beam diameter at the beam waist and $\omega(z)$ is the beam diameter at a $z[cm]$ coordinate defined by

$$\omega(z) = \omega_0 \sqrt{1 + \frac{z^2}{z_0^2}} . \tag{5.2}$$

$$\omega = \omega_0 \sqrt{1 + \left[\frac{z - z_\omega}{z_R}\right]^2} \tag{5.3}$$

$$\omega = \sqrt{\frac{1}{\frac{i}{q}} \cdot \frac{\lambda}{\pi}} . \tag{5.4}$$

The Raleigh range z_0 can be define as

$$z_0 = \frac{\pi\omega_0^2}{\lambda} = \frac{\pi\omega_0^2}{\lambda} . \tag{5.5}$$

The BTEC model follows the Milonni and Eberlys *LASERS* [4]. The "Entry" matrix in figure 5.2 is added to account for the distance from the beam waist that the geometric optic is placed. This allows for optics to be placed at any arbitrary position along the optical axis. This figure also shows matrix multiplication is evaluated from right to left.

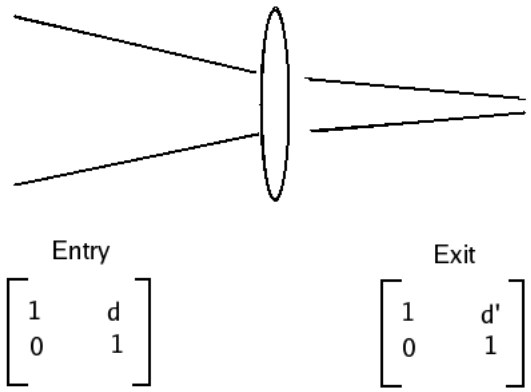

Figure 5.1: Matrix Thin Lens Formulation

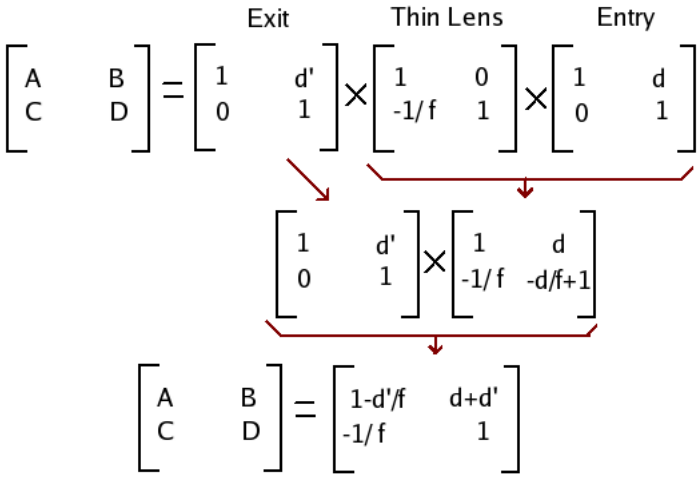

Figure 5.2: ABCD Matrix Breakdown

The components for ABCD matrix are placed into the equation

$$q_f = \frac{Aq_i + B}{-Cq_i + D} \tag{5.6}$$

and then it is set equal to the definition of q_f. Equation 5.6, and the real and imaginary parts of both sides are equated:

$$q_f = \frac{1}{\frac{1}{R} + \frac{i\lambda}{\pi\omega^2}} = \frac{1}{\frac{1}{R} + \frac{i\lambda}{\pi\omega^2}} \tag{5.7}$$

$$\frac{1}{q(z)} = \frac{1}{R(z)} - \frac{2}{ik}\frac{1}{\omega^2(z)} \tag{5.8}$$

$$k = \frac{2\pi}{\lambda} \tag{5.9}$$

The new beam waist can be found by determining where the radius of curvature of the beam approaches infinity. By substituting this value back into the equation for ω(d), the relationship for ω_0 can be found:

$$\omega_0^{'} = \omega_0 \sqrt{\frac{z_R^{'}}{z_R}} = \omega_0 \sqrt{\frac{\frac{z_R}{(Cz_0-D)^2+(C^2 z_R^2)}}{z_R}} \tag{5.10}$$

Once the new spot size is known, the new Rayleigh range can easily be obtained:

$$z_0^{'} = \frac{-(Az_0 - B)(Cz_0 - D) + (ACz_R)}{(Cz_0 - D)^2 + (C^2 z_R^2)} \tag{5.11}$$

$$z_R^{'} = \frac{z_R}{(Cz_0 - D)^2 + (C^2 z_R^2)} \tag{5.12}$$

By finding a general expression for the new beam waist and Rayleigh range, different optical elements can be accounted for by using their ABCD values:

Planar Surface

$$\begin{bmatrix} 1 & 0 \\ 0 & no/n1 \end{bmatrix}$$

Thin Lens

$$\begin{bmatrix} 1 & 0 \\ -1/f & 1 \end{bmatrix}$$

Spherical Lens

$$\begin{bmatrix} 1 & 0 \\ \frac{-(n1-no)}{n1*R} & no/n1 \end{bmatrix}$$

Figure 5.3: Common ABCD Matrix Representations

Several examples of common usages of this ABCD method are shown in figure 5.3 and their parameters are outlined below:

`Planar Bound Propagation`

$$\begin{aligned} A &= 1 \\ B &= 0 \\ C &= 0 \\ D &= \frac{n_1}{n_2} \end{aligned}$$

`Thin Lens Propagation`

$$\begin{aligned} A &= 1 \\ B &= 0 \\ C &= -\frac{1}{f} \\ D &= 1 \end{aligned}$$

`Spherical Lens Propagation`

$$
\begin{aligned}
A &= 1 \\
B &= 0 \\
C &= -\frac{n_2 - n_1}{n_2 R_c} \\
D &= 1
\end{aligned}
$$

where R_c is the radius of curvature of the lens.

Chapter 6

Two-Dimensional Helmholtz Wave Equation

6.1 Derivation of the Theory

It is important,before looking into the modeling methods that are used, to review how the theoretical equations were found and what approximations were used. This chapter includes a derivation of the equation used in the FD-BPM (Finite Difference Beam Propagation Model) starting from Maxwell's equations.

6.1.1 Maxwell Equations

The derivation of the wave propagation equation used in the BTEC model starts with the differential forms of Maxwell's equations:

$$\nabla \cdot \mathbf{E} = 0 \tag{6.1}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{6.2}$$

$$\nabla \times \mathbf{E} = -\mu \frac{\partial \mathbf{H}}{\partial t} \tag{6.3}$$

$$\nabla \times \mathbf{H} = \epsilon \frac{\partial \mathbf{E}}{\partial t} \tag{6.4}$$

There is assumed to be no free current. Permittivity and permeability are assumed to be independent of time. This is true on the time scale of light propagation. However, these properties can change on the time scale of the thermal model and are taken into account in the changing refractive index.

To derive the Wave Equation, the curl of Faraday's equation (6.3) is taken:

$$\nabla \times \nabla \times \mathbf{E} = -\nabla \times \mu \frac{\partial \mathbf{H}}{\partial t} \tag{6.5}$$

$$\nabla \times \nabla \times \mathbf{A} = \nabla(\nabla \cdot \mathbf{A}) - \nabla^2 \mathbf{A} \tag{6.6}$$

Using the vector identity 6.6 and equation 6.5:

$$\nabla(\cancelto{0}{\nabla \cdot \mathbf{E}}) - \nabla^2\mathbf{E} = -\mu\frac{\partial}{\partial t}(\nabla \times \mathbf{H}) \tag{6.7}$$

The divergence of **E** goes to zero from equation 6.1. Substituting equation 6.4 for the curl of **H**, the Wave Equation is found.

6.1.2 Wave Equation

$$\nabla^2\mathbf{E}(\mathbf{r},t) = \mu\epsilon\frac{\partial^2\mathbf{E}(\mathbf{r},t)}{\partial t^2} \tag{6.8}$$

This is the Wave Equation. From separation of variables a solution of the form

$$\mathbf{E}(\mathbf{r},t) = \boldsymbol{\mathcal{E}}(\mathbf{r})e^{-i\omega t} \tag{6.9}$$

can be assumed.

6.1.3 Helmholtz Equation

By substituting equation 6.9 into equation 6.8,

$$\nabla^2\boldsymbol{\mathcal{E}}(\mathbf{r}) + \mu\epsilon\omega^2\boldsymbol{\mathcal{E}}(\mathbf{r}) = 0 \tag{6.10}$$

is found. This is the Vector Helmholtz Equation in an arbitrary spatial coordinate system. It is no longer time dependent.

The Scalar Helmholtz Equation is

$$\nabla^2\mathcal{E}(\mathbf{r}) + \mu\epsilon\omega^2\mathcal{E}(\mathbf{r}) = 0\ . \tag{6.11}$$

In cylindrical coordinates the Laplacian is

$$\nabla^2 A = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial A}{\partial r}\right) + \cancelto{0}{\frac{1}{r^2}\frac{\partial^2 A}{\partial\phi^2}} + \frac{\partial^2 A}{\partial z^2}\ . \tag{6.12}$$

Assuming a problem space with azimuthal symmetry, the ϕ derivative can be ignored.

Also, $\mu\epsilon\omega^2 = \frac{\omega^2}{v^2} = k^2 = k_0^2 n^2$ where k_0 is the free space wave number and n is the refractive index of the material. Using this and equation 6.12, the Helmholtz equation 6.11 in cylindrical coordinates is

$$\frac{\partial^2\mathcal{E}}{\partial z^2} + \frac{1}{r}\left(\frac{\partial\mathcal{E}}{\partial r}\right) + \frac{\partial^2\mathcal{E}}{\partial r^2} + k_0^2 n^2\mathcal{E} = 0\ . \tag{6.13}$$

A plane wave solution of the form

$$\mathcal{E}(r,z) = \Psi(r,z)e^{-ik_0 n_0 z} \tag{6.14}$$

is now assumed, where n_0 is the reference index.

Inserting the assumed solution 6.14 into the Helmholtz equation 6.13, the following equation is derived:

$$-\frac{\partial^2\Psi}{\partial z^2} + 2ik_0 n_0\frac{\partial\Psi}{\partial z} = \frac{\partial^2\Psi}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi}{\partial r} + k_0^2\left[n^2(r,z) - n_0^2\right]\Psi \tag{6.15}$$

equation 6.15 is used in two different solutions. The first is a paraxial approximation which assumes there is no strong focusing or defocusing effects. It utilizes an approximation of an exponential operator as a rational operator. This

approximation is known as the Caley approximation. [9] The second numerical solution allows for strong focusing effects and utilizes Padé approximate operators. Both are based off of equation 6.15 and will be discussed in greater detail later.

6.2 Numerical Approach

6.2.1 Finite Differences

The following finite difference approximations are used in the numerical derivations. Finite differences allow the representation of continuous differential equations in discrete computational space. The finite- differences are shown below for both first and second order derivatives. They are shown first for a uniform grid and then for a stretched grid.

Uniform Grid

`First Order`

$$\frac{\partial \Psi}{\partial r} = \frac{\Psi_{j+1} - \Psi_{j-1}}{2\Delta r} \tag{6.16}$$

`Second Order`

$$\frac{\partial^2 \Psi}{\partial r^2} = \frac{\Psi_{j+1} - 2\Psi_j + \Psi_{j-1}}{\Delta r^2} \tag{6.17}$$

Non-Uniform (Stretched) Grid

In order to extend the physical space out into the radial direction without using vast computational space or making the grid spaces about the axis too large (where the highest concentration of points are needed in order to model a beam accurately), a stretched grid is required [3]. The following finite differences are geared to handle the variable spacial step sizes. The notation used is defined below:

$$\Delta r = r_{j+1} - r_{j-1}$$

$$\Delta r_+ = r_{j+1} - r_j$$

$$\Delta r_- = r_j - r_{j-1}$$

`First Order`

$$\frac{\partial \Psi}{\partial r} = \frac{\Psi_{j+1} - \Psi_{j-1}}{\Delta r} \tag{6.18}$$

`Second Order`

$$\frac{\partial^2 \Psi}{\partial r^2} = \left[\frac{\Psi_{j+1} - \Psi_j}{\Delta r_+} - \frac{\Psi_j - \Psi_{j-1}}{\Delta r_-} \right] \frac{2}{\Delta r} \tag{6.19}$$

6.2.2 Boundary Conditions

These boundary conditions apply to both the Caley method and the Padé method. These methods exploit the symmetry of a laser beam to compute a solution at the $r = 0$ boundary. They also allow circumvention of the division by 0 in equation 6.15.

Due to the symmetry of an axial laser beam,

$$\frac{\partial \Psi(r = 0, z)}{\partial r} = 0 \,. \tag{6.20}$$

By evaluation of equation 6.20 using the finite difference representation from equation 6.18, the following relation can be found across the $r = 0$ boundary:

$$\Psi^i_{-1} = \Psi^i_1 \tag{6.21}$$

This is convenient because the Ψ^i_{-1} point doesn't exist in the problem space and is considered a "phantom point." The relation in 6.20 does not, however, imply that $\frac{1}{r}\frac{\partial\Psi(r=0,z)}{\partial r} = 0$ as $\frac{0}{0}$ does not necessarily equate to 0. To find this relationship, L'Hospital's Rule is utilized.

$$\lim_{r\to 0}\frac{1}{r}\frac{\partial\Psi}{\partial r} = \frac{\partial^2\Psi}{\partial r^2} \tag{6.22}$$

Also, because of the symmetry of the problem, the phantom grid spaces are evaluated by

$$\begin{aligned}\Delta r_- &= \Delta r_+\\ \Delta r &= 2\Delta r_+\end{aligned} \tag{6.23}$$

6.3 Caley Method

The method of interest for use in the BTEC thermal model is the Padé Method. The Caley Method was implemented first because it was conceptually easier to understand. However, the Padé Method is a more powerful method as it lacks the paraxial assumption. The implementation of the Caley Method gave results against which the Padé Method implementation could be validated.

In the Caley form solution, a paraxial approximation is made to the wave equation. If it is assumed that there are no strong focusing or defocusing effects occurring in the media, the second axial derivative can be assumed to be negligible and dropped [2]:

$$-2ik_0n_0\frac{\partial\Psi}{\partial z} = \frac{\partial^2\Psi}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi}{\partial r} + k_0^2\left[n^2(r,z) - n_0^2\right]\Psi \tag{6.24}$$

The equation is rearranged for the ease of calculation to yield

$$i\frac{\partial\Psi}{\partial z} = \frac{-1}{2kn_0}\left[\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + k_0^2\left[n^2(r,z) - n_0^2\right]\right]\Psi \tag{6.25}$$

$$i\frac{\partial\Psi}{\partial z} = \hat{S}\,\Psi(z)\,. \tag{6.26}$$

In equation 6.26, an operator has been defined for ease of calculation. This $\hat{S}$ will contain the derivatives in the r direction as well as constants. It is defined as

$$\hat{S} = \frac{-1}{2kn_0}\left[\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + k_0^2\left[n^2(r,z) - n_0^2\right]\right]\,. \tag{6.27}$$

The solution to equation 6.26 is found by separating and integrating. With appropriately chosen bounds, the solution appears as

$$\Psi(r, z+\Delta z) = e^{-i\Delta z\hat{S}}\,\Psi(r,z)\,. \tag{6.28}$$

6.3.1 Caley's Form of the Exponential Operator

The Caley method as it is referred to here is a Crank-Nicholson Method that uses the Caley form of the exponential operator. The Caley form is an approximation that allows the exponential operator to be represented as a rational operator:

$$e^{-i\hat{S}\,\Delta z} \simeq \frac{1 - \frac{1}{2}i\hat{S}\,\Delta z}{1 + \frac{1}{2}i\hat{S}\,\Delta z} \tag{6.29}$$

Using the Caley relationship 6.29, equation 6.28 becomes

$$\left[1 + \frac{i\hat{S}\,\Delta z}{2}\right]\Psi(z + \Delta z) = \left[1 - \frac{i\hat{S}\,\Delta z}{2}\right]\Psi(z)\,. \tag{6.30}$$

Substituting the transverse derivatives and constants contained in the $\hat{S}$, from equation 6.27 and the finite difference representations of these transverse derivatives from Equations 6.18 and 6.19, equation 6.30 becomes Equations 6.31 and 6.32. Equation 6.31 is the left side of equation 6.30 and 6.32 is the right side:

LHS

$$\Psi_j^{i+1} + \frac{-i\Delta z}{4k_0 n_0}\left[\left(\frac{\Psi_{j+1}^{i+1} - \Psi_j^{i+1}}{\Delta r_+} - \frac{\Psi_j^{i+1} - \Psi_{j-1}^{i+1}}{\Delta r_-}\right)\frac{2}{\Delta r} + \frac{1}{r}\frac{\Psi_{j+1}^{i+1} - \Psi_{j-1}^{i+1}}{\Delta r} + k_0^2\left[n^2(r,z) - n_0^2\right]\Psi_j^{i+1}\right] \tag{6.31}$$

RHS

$$\Psi_j^{i} - \frac{-i\Delta z}{4k_0 n_0}\left[\left(\frac{\Psi_{j+1}^{i} - \Psi_j^{i}}{\Delta r_+} - \frac{\Psi_j^{i} - \Psi_{j-1}^{i}}{\Delta r_-}\right)\frac{2}{\Delta r} + \frac{1}{r}\frac{\Psi_{j+1}^{i} - \Psi_{j-1}^{i}}{\Delta r} + k_0^2\left[n^2(r,z) - n_0^2\right]\Psi_j^{i}\right]\,, \tag{6.32}$$

where the following notation is again adapted:

$$\Psi(r,z) = \Psi_j^i$$

$$\Psi(r - \Delta r, z + \Delta z) = \Psi_{j-1}^{i+1}$$

6.3.2 System of Equations

In order to easily view the system of equations formed in 6.31 and 6.32, the equations are ordered by the Ψ values at each point in the r and z directions.

LHS

$$\begin{aligned}
&\Psi_{j-1}^{i+1}\left[-\frac{i\Delta z}{4k_0 n_0}\left(\frac{2}{\Delta r_- \Delta r} - \frac{1}{r\Delta r}\right)\right] + \\
&\Psi_{j}^{i+1}\left[1 + \frac{i\Delta z}{4k_0 n_0}\left(\frac{2}{\Delta r_+ \Delta r} + \frac{2}{\Delta r_- \Delta r} - k_0^2 n + k_0^2 n_0^2\right)\right] + \\
&\Psi_{j+1}^{i+1}\left[-\frac{i\Delta z}{4k_0 n_0}\left(\frac{2}{\Delta r_+ \Delta r} + \frac{1}{r\Delta r}\right)\right]
\end{aligned} \tag{6.33}$$

RHS

$$\Psi^i_{j-1}\left[\frac{i\Delta z}{4k_0n_0}\left(\frac{2}{\Delta r_-\Delta r}-\frac{1}{r\Delta r}\right)\right]+$$
$$\Psi^i_j\left[1-\frac{i\Delta z}{4k_0n_0}\left(\frac{2}{\Delta r_+\Delta r}+\frac{2}{\Delta r_-\Delta r}-k_0^2n+k_0^2n_0^2\right)\right]+ \tag{6.34}$$
$$\Psi^i_{j+1}\left[\frac{i\Delta z}{4k_0n_0}\left(\frac{2}{\Delta r_+\Delta r}+\frac{1}{r\Delta r}\right)\right]$$

This is commonly referred to as a tridiagonal matrix system of equations because it can be represented in matrix form shown in section 6.5.1.

6.3.3 Boundary Conditions

Equation 6.15 is undefined at $r = 0$. To circumvent this, axial equations can be added to the system of equations by applying appropriate boundary conditions to the problem. Applying the boundary conditions from equation 6.22, $\hat{S_A}$ (the A subscript denotes the axial solution) becomes

$$\hat{S}_A = \frac{1}{2k}\left[\frac{2\partial^2}{\partial r^2}+k_0^2\left[n^2(r,z)-n_0^2\right]\right]. \tag{6.35}$$

Using equation 6.30 and the new operator $\hat{S}_A$, the numerical equation at the axial boundary becomes the following:

LHS

$$\Psi^{i+1}_{-1}\left[-\frac{i\Delta z}{4k_0n_0}\left(\frac{2}{\Delta r_+^2}\right)\right]+$$
$$\Psi^{i+1}_0\left[1+\frac{i\Delta z}{4k_0n_0}\left(\frac{4}{\Delta r_+^2}-k_0^2n+k_0^2n_0^2\right)\right]+ \tag{6.36}$$
$$\Psi^{i+1}_1\left[-\frac{i\Delta z}{4k_0n_0}\left(\frac{2}{\Delta r_+^2}\right)\right]$$

RHS

$$\Psi^{i}_{-1}\left[\frac{i\Delta z}{4k_0n_0}\left(\frac{2}{\Delta r_+^2}\right)\right]+$$
$$\Psi^{i}_0\left[1-\frac{i\Delta z}{4k_0n_0}\left(\frac{4}{\Delta r_+^2}-k_0^2n+k_0^2n_0^2\right)\right]+ \tag{6.37}$$
$$\Psi^{i}_1\left[\frac{i\Delta z}{4k_0n_0}\left(\frac{2}{\Delta r_+^2}\right)\right]$$

Applying the relationship for the "phantom point", Ψ^i_{-1}, from equation 6.21, Equations 6.36 and 6.37 become as shown below:

RHS

$$\Psi^{i}_0\left[1-\frac{i\Delta z}{4k_0n_0}\left(\frac{4}{\Delta r_+^2}-k_0^2n+k_0^2n_0^2\right)\right]+$$
$$\Psi^{i}_1\left[\frac{i\Delta z}{4k_0n_0}\left(\frac{4}{\Delta r_+^2}\right)\right] \tag{6.38}$$

LHS

$$\Psi_0^{i+1}\left[1+\frac{i\Delta z}{4k_0n_0}\left(\frac{4}{\Delta r_+^2}-k_0^2n+k_0^2n_0^2\right)\right]+ \\ \Psi_1^{i+1}\left[-\frac{i\Delta z}{4k_0n_0}\left(\frac{4}{\Delta r_+^2}\right)\right] \tag{6.39}$$

The system of equations created can be represented in the matrix form shown in section 6.5.1.

6.4 Padé Method

The Padé method starts with equation 6.15 as well. In this case, a paraxial solution is not assumed and the second derivative in the axial direction is left intact. Equation 6.15 is rearranged into the recursive form shown in equation 6.40 to become

$$\frac{\partial\Psi}{\partial z}=-\frac{\frac{i\hat{P}}{2k_0n_0}}{1+\frac{i}{2k_0n_0}\frac{\partial}{\partial z}}\Psi \tag{6.40}$$

where

$$\hat{P}=\left[\frac{\partial^2}{\partial r^2}+\frac{1}{r}\frac{\partial}{\partial r}+k_0^2\left[n^2(r,z)-n_0^2\right]\right]. \tag{6.41}$$

6.4.1 Padé Approximate

The recursive nature of this equation along with the help of [1] yields the Padé(1,1) form solution in Equation 6.42:

$$\frac{\partial\Psi}{\partial z}=-\frac{\frac{i\hat{P}}{2k_0n_0}}{1+\frac{\hat{P}}{4k_0^2n_0^2}}\Psi \tag{6.42}$$

Higher-order Padé approximates are available such as the Padé(3,3). The Padé(1,1) derivation will result in solving a tridiagonal matrix system of equations. The higher-order approximates result in solving more complex systems of equations such as a pentadiagonal matrix system of equations.

Using a finite difference approximation to the derivative with respect to z on the left hand side and taking an average of the Ψ on the right hand side:

$$\frac{\Psi^{i+1}-\Psi^i}{\Delta z}=-\frac{\frac{i\hat{P}}{2k_0n_0}}{1+\frac{\hat{P}}{4k_0^2n_0^2}}\frac{\Psi^{i+1}+\Psi^i}{2} \tag{6.43}$$

By rearranging equation 6.43, equation 6.44 can be obtained:

$$\hat{P}\Psi^{i+1}+ikn_0\Delta z\hat{P}\Psi^{i+1}+4k_0^2n_0^2\Psi^{i+1}=\hat{P}\Psi^i-ikn_0\Delta z\hat{P}\Psi^i+4k_0^2n_0^2\Psi^i \tag{6.44}$$

Substituting the r derivatives and constants contained in the $\hat{P}$ operator, Equations 6.45 and 6.46 can be found. equation 6.45 is the left side of equation 6.44 and equation 6.46 is the right side of equation 6.44:

LHS

$$\begin{aligned}
&\frac{\partial^2\Psi^{i+1}}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi^{i+1}}{\partial r} + k_0^2\left[n^2 - n_0^2\right]\Psi^{i+1} \\
&+ ikn_0\Delta z\left[\frac{\partial^2\Psi^{i+1}}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi^{i+1}}{\partial r} + k_0^2\left[n^2 - n_0^2\right]\Psi^{i+1}\right] \\
&+ 4k_0^2n_0^2\Psi^{i+1}
\end{aligned} \tag{6.45}$$

RHS

$$\begin{aligned}
&\frac{\partial^2\Psi^{i}}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi^{i}}{\partial r} + k_0^2\left[n^2 - n_0^2\right]\Psi^{i} \\
&- ikn_0\Delta z\left[\frac{\partial^2\Psi^{i}}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi^{i}}{\partial r} + k_0^2\left[n^2 - n_0^2\right]\Psi^{i}\right] \\
&+ 4k_0^2n_0^2\Psi^{i+1}
\end{aligned} \tag{6.46}$$

Applying finite difference approximations from Equations 6.18 and 6.19 for the transverse derivatives, Equations 6.45 and 6.46 become as shown:

LHS

$$\begin{aligned}
&\frac{2}{\Delta r}\left[\frac{\Psi_{j+1}^{i+1} - \Psi_j^{i+1}}{\Delta r_+} - \frac{\Psi_j^{i+1} - \Psi_{j-1}^{i+1}}{\Delta r_-}\right] + \frac{1}{r}\left[\frac{\Psi_{j+1}^{i+1} - \Psi_{j-1}^{i+1}}{\Delta r}\right] + k_0^2\left[n^2 - n_0^2\right]\Psi_j^{i+1} \\
&+ ikn_0\Delta z\left(\frac{2}{\Delta r}\left[\frac{\Psi_{j+1}^{i+1} - \Psi_j^{i+1}}{\Delta r_+} - \frac{\Psi_j^{i+1} - \Psi_{j-1}^{i+1}}{\Delta r_-}\right] + \frac{1}{r}\left[\frac{\Psi_{j+1}^{i+1} - \Psi_{j-1}^{i+1}}{\Delta r}\right] + k_0^2\left[n^2 - n_0^2\right]\Psi_j^{i+1}\right) \\
&+ 4k_0^2n_0^2\Psi^{i+1}
\end{aligned} \tag{6.47}$$

RHS

$$\begin{aligned}
&\frac{2}{\Delta r}\left[\frac{\Psi_{j+1}^{i} - \Psi_j^{i}}{\Delta r_+} - \frac{\Psi_j^{i} - \Psi_{j-1}^{i}}{\Delta r_-}\right] + \frac{1}{r}\left[\frac{\Psi_{j+1}^{i} - \Psi_{j-1}^{i}}{\Delta r}\right] + k_0^2\left[n^2 - n_0^2\right]\Psi_j^{i} \\
&- ikn_0\Delta z\left(\frac{2}{\Delta r}\left[\frac{\Psi_{j+1}^{i} - \Psi_j^{i}}{\Delta r_+} - \frac{\Psi_j^{i} - \Psi_{j-1}^{i}}{\Delta r_-}\right] + \frac{1}{r}\left[\frac{\Psi_{j+1}^{i} - \Psi_{j-1}^{i}}{\Delta r}\right] + k_0^2\left[n^2 - n_0^2\right]\Psi_j^{i}\right) \\
&+ 4k_0^2n_0^2\Psi^{i}
\end{aligned} \tag{6.48}$$

By rearranging and multiplying by Δr, Equations 6.47 and 6.48 become the equations that follow:

LHS

$$\begin{aligned}
&\frac{2\Psi_{j-1}^{i+1}}{\Delta r_-} - \frac{\Psi_{j-1}^{i+1}}{r} + \frac{2ik_0n_0\Delta z\Psi_{j-1}^{i+1}}{\Delta r_-} - \frac{ik_0n_0\Delta z\Psi_{j-1}^{i+1}}{r} \\
&- \frac{2\Psi_j^{i+1}}{\Delta r_+} - \frac{2\Psi_j^{i+1}}{\Delta r_-} + k_0^2\Delta r\left[n^2 - n_0^2\right]\Psi_j^{i+1} \\
&- \frac{2ik_0n_0\Delta z\Psi_j^{i+1}}{\Delta r_+} - \frac{2ik_0n_0\Delta z\Psi_j^{i+1}}{\Delta r_-} + ik_0n_0\Delta zk_0^2\Delta r\left[n^2 - n_0^2\right]\Psi_j^{i+1} \\
&+ 4k_0^2n_0^2\Delta r\Psi_j^{i+1} \\
&+ \frac{2\Psi_{j+1}^{i+1}}{\Delta r_+} + \frac{\Psi_{j+1}^{i+1}}{r} + \frac{2ik_0n_0\Delta z\Psi_{j+1}^{i+1}}{\Delta r_+} + \frac{ik_0n_0\Delta z\Psi_{j+1}^{i+1}}{r}
\end{aligned} \tag{6.49}$$

RHS

$$\begin{aligned}
&\frac{2\Psi^i_{j-1}}{\Delta r_-} - \frac{\Psi^i_{j-1}}{r} - \frac{2ik_0n_0\Delta z\Psi^i_{j-1}}{\Delta r_-} + \frac{ik_0n_0\Delta z\Psi^i_{j-1}}{r} \\
&- \frac{2\Psi^{i+1}_j}{\Delta r_+} - \frac{2\Psi^{i+1}_j}{\Delta r_-} + k_0^2\Delta r\left[n^2 - n_0^2\right]\Psi^{i+1}_j \\
&+ \frac{2ik_0n_0\Delta z\Psi^i_j}{\Delta r_+} + \frac{2ik_0n_0\Delta z\Psi^i_j}{\Delta r_-} - ik_0n_0\Delta zk_0^2\Delta r\left[n^2 - n_0^2\right]\Psi^i_j \\
&+ 4k_0^2n_0^2\Delta r\Psi^i_j \\
&+ \frac{2\Psi^i_{j+1}}{\Delta r_+} + \frac{\Psi^i_{j+1}}{r} - \frac{2ik_0n_0\Delta z\Psi^i_{j+1}}{\Delta r_+} - \frac{ik_0n_0\Delta z\Psi^i_{j+1}}{r}
\end{aligned} \tag{6.50}$$

The solutions at every r position are then factored out of Equations 6.47 and 6.50:

LHS

$$\begin{aligned}
&\left(\frac{2}{\Delta r_-} - \frac{1}{r}\right)(1 + ik_0n_0\Delta z)\,\Psi^{i+1}_{j-1} \\
&+ \left[\left(-\frac{2}{\Delta r_+} - \frac{2}{\Delta r_-} + k_0^2\Delta r\left[n^2 - n_0^2\right]\right)(1 + ik_0n_0\Delta z) + 4k_0^2n_0^2\Delta r\right]\Psi^{i+1}_j \\
&+ \left(\frac{2}{\Delta r_+} + \frac{1}{r}\right)(1 + ik_0n_0\Delta z)\,\Psi^{i+1}_{j+1}
\end{aligned} \tag{6.51}$$

RHS

$$\begin{aligned}
&\left(\frac{2}{\Delta r_-} - \frac{1}{r}\right)(1 - ik_0n_0\Delta z)\,\Psi^{i}_{j-1} \\
&+ \left[\left(-\frac{2}{\Delta r_+} - \frac{2}{\Delta r_-} + k_0^2\Delta r\left[n^2 - n_0^2\right]\right)(1 - ik_0n_0\Delta z) + 4k_0^2n_0^2\Delta r\right]\Psi^{i}_j \\
&+ \left(\frac{2}{\Delta r_+} + \frac{1}{r}\right)(1 - ik_0n_0\Delta z)\,\Psi^{i}_{j+1}
\end{aligned} \tag{6.52}$$

6.4.2 System of Equations

The system of equations in 6.51 and 6.52 can be rewritten as

$$\begin{aligned}
&\alpha(-)\beta(+)\Psi^{i+1}_{j-1} + \left[\gamma\beta(+) + 4k_0^2n_0^2\Delta r\right]\Psi^{i+1}_j + \alpha(+)\beta(+)\Psi^{i+1}_{j+1} \\
&= \alpha(-)\beta(-)\Psi^{i}_{j-1} + \left[\gamma\beta(-) + 4k_0^2n_0^2\Delta r\right]\Psi^{i}_j + \alpha(+)\beta(-)\Psi^{i}_{j+1}
\end{aligned} \tag{6.53}$$

where

$$\begin{aligned}
\alpha(\pm) &= \frac{2}{\Delta r_{(\pm)}} \pm \frac{1}{r} \\
\beta(\pm) &= 1 \pm ik_0n_0\Delta z \\
\gamma &= -\frac{2}{\Delta r_-} - \frac{2}{\Delta r_+} + k_0^2\Delta r\left[n^2 - n_0^2\right] .
\end{aligned}$$

6.4.3 Boundary Conditions

At $r = 0$, it is again necessary to use the boundary conditions. Applying equation 6.22, the operator $\hat{P}_A$ becomes

$$\hat{P}_A = \left[\frac{2\partial^2}{\partial r^2} + k_0^2\left[n^2(r,z) - n_0^2\right]\right] .$$

Applying equation 6.4.3 makes equation 6.44 become:

$$\hat{P}_A\Psi^{i+1} + ikn_0\Delta z\hat{P}_A\Psi^{i+1} + 4k_0^2n_0^2\Psi^{i+1} = \hat{P}_A\Psi^i - ikn_0\Delta z\hat{P}_A\Psi^i + 4k_0^2n_0^2\Psi^i . \tag{6.54}$$

Following the same steps as for the rest of the grid points, the numerical equation at the $r = 0$ boundary becomes

$$\begin{aligned}&\left[\gamma\beta(+) + 4k_0^2n_0^2\Delta r\right]\Psi_0^{i+1} + \alpha(+)\beta(+)\Psi_1^{i+1}\\ &= +\left[\gamma\beta(-) + 4k_0^2n_0^2\Delta r\right]\Psi_0^i + \alpha(+)\beta(-)\Psi_1^i\end{aligned} \tag{6.55}$$

where

$$\begin{aligned}\alpha(\pm) &= \frac{8}{\Delta r_{(\pm)}}\\ \beta(\pm) &= 1 \pm ik_0n_0\Delta z\\ \gamma &= -\frac{4}{\Delta r_-} - \frac{4}{\Delta r_+} + k_0^2\Delta r\left[n^2 - n_0^2\right] .\end{aligned}$$

Again this system of equations can be written in the matrix form found in section 6.5.1.

6.5 Solving the Tridiagonal Matrix Equation

6.5.1 Tridiagonal Matrix Equation

The system of equations found in the Caley method and the system of equations found in the Padé method can be represented in the following tridiagonal matrix form. The matrix values are the terms multiplied by the Ψ at each grid point. These matrix values are different for each method:

$$\begin{bmatrix} b_0 & c_0 & 0 & & \cdots & & 0\\ a_1 & b_1 & c_1 & & & & \\ 0 & a_2 & b_2 & c_2 & & & \vdots\\ & & \ddots & \ddots & \ddots & & \\ \vdots & & & \ddots & \ddots & \ddots & 0\\ & & & & a_{M-2} & b_{M-2} & c_{M-2}\\ 0 & & \cdots & & 0 & a_{M-1} & b_{M-1}\end{bmatrix}\begin{bmatrix}\Psi_1^i\\ \Psi_2^i\\ \Psi_3^i\\ \vdots\\ \vdots\\ \Psi_{M-2}^i\\ \Psi_{M-1}^i\end{bmatrix} = \begin{bmatrix} b'_0 & c'_0 & 0 & & \cdots & & 0\\ a'_1 & b'_1 & c'_1 & & & & \\ 0 & a'_2 & b'_2 & c'_2 & & & \vdots\\ & & \ddots & \ddots & \ddots & & \\ \vdots & & & \ddots & \ddots & \ddots & 0\\ & & & & a'_{M-2} & b'_{M-2} & c'_{M-2}\\ 0 & & \cdots & & 0 & a'_{M-1} & b'_{M-1}\end{bmatrix}\begin{bmatrix}\Psi_1^{i+1}\\ \Psi_2^{i+1}\\ \Psi_3^{i+1}\\ \vdots\\ \vdots\\ \Psi_{M-2}^{i+1}\\ \Psi_{M-1}^{i+1}\end{bmatrix}$$

This matrix can be represented in the matrix-vector notation as

$$\mathbb{A}\boldsymbol{\Psi}^i = \mathbb{B}\boldsymbol{\Psi}^{i+1} . \tag{6.56}$$

6.5.2 Explicit Solution

The left side of equation 6.56 is made of known information. That side of the equation can be solved explicitly. The computer directly multiplies all the matrix elements with the known solution and equation 6.56 becomes

$$\mathbf{\Phi} = \mathbb{B}\mathbf{\Psi}^{i+1} \ . \tag{6.57}$$

Where $\mathbf{\Phi}$ is the array result of explicitly solving the left side of equation 6.56.

6.5.3 Thomas Algorithm

Equation 6.57 can not be solved explicitly. The inverse of the matrix $\mathbb{B}$ could be taken and then explicitly multiplied with array components in $\mathbf{\Phi}$ to find $\mathbf{\Psi}$, but this is computationally intensive. Instead, the Thomas algorithm [9] is utilized. The Thomas algorithm is a fast, efficient way of solving this equation without finding an inverse. "Numerical Recipes" contains an algorithm called tridiag that was used for this step.

6.6 Propagation using Hankel Transforms

6.6.1 Bessel Functions

Over an infinite range, $[0, \infty]$, the Bessel functions form a complete set. Any arbitrary function can therefore be represented as a linear combination of Bessel functions. The set of Bessel functions can be written as abstract ket vectors, $|J_{p,\nu}\rangle$, of the vector space whose inner product is defined as[1]:

$$\langle g|f\rangle = 2\pi \int_0^a f(r)\, g(r)\, r dr \tag{6.58}$$

The set $|J_{p,\nu}\rangle$ span the inner product space, or form a complete basis. The index p denotes the Bessel function order, and ν is a continuous index. Casting this into the coordinate basis gives

$$\langle r|J_{p,\nu}\rangle \rightarrow J_p(2\pi\nu r) \ .$$

Over a finite range $[0, R]$, the continuous index is quantized. ($\nu \rightarrow n$). The complete set over the finite range becomes

$$\langle r|J_{p,n}\rangle \rightarrow J_p\left(\tfrac{\alpha_{p,n}}{R} r\right) \qquad n = 1, 2, \ldots \ ,$$

where $\alpha_{p,n}$ denotes the nth index of the pth order Bessel function. The Bessel functions are convenient to work with because they are the eigenfunctions of the radial component of the cylindrical Laplace operator:

$$\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial}{\partial r}\right)\right] J_{p,n}\left(\frac{\alpha_{p,n}}{R} r\right) = -\frac{\alpha_{p,n}^2}{R^2} J_p\left(\frac{\alpha_{p,n}}{R} r\right)$$

or, written abstractly...

$$\mathbb{D}^{(r)}|J_{p,n}\rangle = -\frac{\alpha_{p,n}^2}{R^2}|J_{p,n}\rangle$$

A few properties of the Bessel functions will be useful in the following derivation. Note that in the notation used here, $\langle \frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle$ corresponds to $J_{p+1}\left(\alpha_{p,n}\right)$

Closure (Continuous):

$$\langle J_{p,\nu}|J_{p,\nu'}\rangle = \frac{1}{\nu}\delta(\nu - \nu') \tag{6.59}$$

[1] The 2π here is included with the cylindrical coordinates in mind.

Orthogonality (Finite):

$$\langle J_{p,n}|J_{p,m}\rangle = 0 \tag{6.60}$$

Normalization (Finite):

$$\langle J_{p,n}|J_{p,n}\rangle = \frac{R^2}{2}\left[\langle\frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle\right]^2 \tag{6.61}$$

6.6.2 Projections

The projection operators of the Bessel basis and the coordinate basis are:

$$\mathbb{P}^{(J_\nu)} = \frac{|J_{p,\nu}\rangle\langle J_{p,\nu}|}{\langle J_{p,\nu}|J_{p,\nu}\rangle}$$
$$\mathbb{P}^{(r)} = \frac{|r\rangle\langle r|}{\langle r|r\rangle}$$

A vector in the space can be represented by it's components in the Bessel, or coordinate basis by:

$$|g\rangle = \int_0^\infty \mathbb{P}^{(J_\nu)}|g\rangle d\nu$$
$$|g\rangle = \int_0^\infty \mathbb{P}^{(r)}|g\rangle dr$$

Note here that over the infinite range, we have

$$\langle r|r\rangle = \frac{\delta(r-r')}{2\pi r}$$
$$\langle J_{p,\nu}|J_{p,\nu'}\rangle = \frac{\delta(\nu-\nu')}{2\pi\nu}$$

from the Closure equation (equation 6.59) and the definition of the inner product (equation 6.58), and the projections of g can be written

$$|g\rangle = 2\pi\int_0^\infty |J_{p,\nu}\rangle\langle J_{p,\nu}|g\rangle\nu d\nu$$
$$|g\rangle = 2\pi\int_0^\infty |r\rangle\langle r|g\rangle r dr\ .$$

Over a finite range, the continuous index, ν, on the Bessel functions becomes discrete, and the integral will go to a sum. The coordinate basis vectors are still indexed by a continuous variable.

$$|g\rangle = \sum_{n=1}^{\infty}\mathbb{P}^{(J_n)}|g\rangle$$
$$|g\rangle = \int_0^R \mathbb{P}^{(r)}|g\rangle$$

Again, from the inner product definition and normalization integral (equation 6.61), these can be written

$$|g\rangle = \sum_{n=1}^{\infty}\frac{R^2}{2}\left[\langle\frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle\right]^2 J_n|g\rangle$$
$$|g\rangle = 2\pi\int_0^R |r\rangle\langle r|g\rangle r dr\ .$$

From these representations, it is obvious that the sum/integral over projection operators is the identity matrix.
Two approximations will be used concerning the sum of the projection operators. The first is that of the coordinate basis projection operators over a finite range. The operators must be integrated over to give the identity matrix, but can be approximated by a sum:

$$\int_0^R \frac{|r\rangle\langle r|}{\langle r|r\rangle} \equiv \mathbb{I} \approx \sum_{m=0}^{\infty} \frac{|r_m\rangle\langle r_m|}{\langle r_m|r_m\rangle} \tag{6.62}$$

Next, we note that if the index, ν, has a finite bound, V, then the infinite limit of the integral can be replaced with V:

$$\int_0^\infty \frac{|J_{p,\nu}\rangle\langle J_{p,\nu}|}{\langle J_{p,\nu}|J_{p,\nu}\rangle} \equiv \mathbb{I} \approx \int_0^V \frac{|J_{p,\nu}\rangle\langle J_{p,\nu}|}{\langle J_{p,\nu}|J_{p,\nu}\rangle} \tag{6.63}$$

These will be used as approximations to the identity matrix.

6.6.3 The Quasi-Discrete Hankel Transform

The beam profile is assumed to be an azimuthally symmetric, separable, function:

$$\Psi(r,\theta,z) \to \Psi(r,z) \to \Psi^{(r)}(r)\,\Psi^{(z)}(z)$$

The electric field distribution, and again the Bessel functions, can be written as abstract ket vectors: [2]

$$|\Psi\rangle$$
$$|J_{p,n}\rangle$$

Casting these vectors onto the coordinate basis retrieves the functional representation

$$\langle r,z|\Psi\rangle \to \Psi(r,z)$$
$$\langle r|J_{p,n}\rangle \to J_p\left(\frac{\alpha_{p,n}}{R}r\right).$$

If R is chosen to be larger than the beam radius, the radial electric field distribution can be written as

$$\langle r|\Psi\rangle = \langle r|\sum_{n=1}^{\infty} \frac{|J_{p,n}\rangle\langle J_{p,n}|}{\langle J_{p,n}|J_{p,n}\rangle}|\Psi\rangle\,, \tag{6.64}$$

where the sum of the projection operators has been inserted. Slight rearranging of terms gives

$$\langle r|\Psi\rangle = \sum_{n=1}^{\infty} \frac{\langle r|J_{p,n}\rangle}{\langle J_{p,n}|J_{p,n}\rangle}\langle J_{p,n}|\Psi\rangle\,. \tag{6.65}$$

It is apparent that the sum is a Bessel series expansion of the electric field distribution. The inner product $\langle J_{p,n}|\Psi\rangle$ is the electric field distribution represented in the Bessel basis, and is the Hankel Transform of the original electric field distribution represented in the coordinate basis, $\langle r|\Psi\rangle$. Similarly, the Hankel Transform can be expanded in the coordinate basis:

$$\langle J_{p,n}|\Psi\rangle = \int_0^R \frac{\langle J_{p,n}|r\rangle\langle r|\Psi\rangle}{\langle r|r\rangle}$$

Using equation 6.62 to replace the integration with a sum and rearranging terms gives

$$\langle J_{p,n}|\Psi\rangle = \sum_{m=1}^{\infty} \frac{\langle J_{p,n}|r_m\rangle}{\langle r_m|r_m\rangle}\langle r_m|\Psi\rangle\,. \tag{6.66}$$

[2] The Bessel functions are assumed to be functions of the r coordinate

Over a finite range, equation 6.66 is an approximation of the Hankel Transform, while equation 6.65 is the Inverse Hankel Transform. A means of numerically evaluating these transforms can be reached through a symmetric derivation.

Hankel Transform	Inverse Hankel Transform
$\langle r\|\Psi\rangle = \sum_{n=1}^{\infty} \frac{\langle r\|J_{p,n}\rangle}{\langle J_{p,n}\|J_{p,n}\rangle}\langle J_{p,n}\|\Psi\rangle$	$\langle J_{p,n}\|\Psi\rangle = \sum_{m=1}^{\infty} \frac{\langle J_{p,n}\|r_m\rangle}{\langle r_m\|r_m\rangle}\langle r_m\|\Psi\rangle$
expand the denominator with projection operators	
$\sum_{n=1}^{\infty} \frac{\langle r\|J_{p,n}\rangle}{2\pi\int_0^R \langle J_{p,n}\|r\rangle\langle r\|J_{p,n}\rangle r dr}\langle J_{p,n}\|\Psi\rangle$	$\sum_{m=1}^{\infty} \frac{\langle J_{p,n}\|r_m\rangle}{2\pi\int_0^V \langle r_m\|J_{p,\nu}\rangle\langle J_{p,\nu}\|r_m\rangle \nu d\nu}\langle r_m\|\Psi\rangle$
Now, if the r_m's are chosen to be $\frac{\alpha_{p,n}}{V}$, the two integrals in the denominators reduce to the normalization equation (equation 6.61)	
$\sum_{n=1}^{\infty} \frac{\langle r\|J_{p,n}\rangle}{2\pi\frac{R^2}{2}\left[\langle\frac{\alpha_{p,n}}{2\pi}\|J_{p+1,\nu=1}\rangle\right]^2}\langle J_{p,n}\|\Psi\rangle$	$\sum_{m=1}^{\infty} \frac{\langle J_{p,n}\|r_m\rangle}{2\pi\frac{V^2}{2}\left[\langle\frac{\alpha_{p,m}}{2\pi}\|J_{p+1,\nu=1}\rangle\right]^2}\langle r_m\|\Psi\rangle$
Evaluating the Hankel Transform at the same r coordinates chosen for the Inverse Hankel Transform, gives each function in terms of the other	
$\langle r_m\|\Psi\rangle = \sum_{n=1}^{\infty} \frac{2\langle r_m\|J_{p,n}\rangle}{2\pi R^2\left[\langle\frac{\alpha_{p,n}}{2\pi}\|J_{p+1,\nu=1}\rangle\right]^2}\langle J_{p,n}\|\Psi\rangle$	$\langle J_{p,n}\|\Psi\rangle = \sum_{m=1}^{\infty} \frac{2\langle J_{p,n}\|r_m\rangle}{2\pi V^2\left[\langle\frac{\alpha_{p,m}}{2\pi}\|J_{p+1,\nu=1}\rangle\right]^2}\langle r_m\|\Psi\rangle$

With a little algebra, the two representation can be written in a symmetric form.

$$\frac{R\langle r_m|\Psi\rangle}{\left|\langle\frac{\alpha_{p,m}}{2\pi}|J_{p+1,\nu=1}\rangle\right|} = \sum_{n=1}^{\infty} \frac{2\langle r_m|J_{p,n}\rangle}{2\pi RV\left|\langle\frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle\right|\left|\langle\frac{\alpha_{p,m}}{2\pi}|J_{p+1,\nu=1}\rangle\right|}\frac{V\langle J_{p,n}|\Psi\rangle}{\left|\langle J_{p+1,\nu=1}|\frac{\alpha_{p,n}}{2\pi}\rangle\right|}$$

$$\frac{V\langle J_{p,n}|\Psi\rangle}{\left|\langle\frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle\right|} = \sum_{n=1}^{\infty} \frac{2\langle J_{p,n}|r_m\rangle}{2\pi VR\left|\langle\frac{\alpha_{p,m}}{2\pi}|J_{p+1,\nu=1}\rangle\right|\left|\langle\frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle\right|}\frac{R\langle r_m|\Psi\rangle}{\left|\langle J_{p+1,\nu=1}|\frac{\alpha_{p,m}}{2\pi}\rangle\right|}$$

We define two vectors and a matrix,

$$\Phi_m^{(r)} = \frac{R\langle r_m|\Psi\rangle}{\left|\langle \frac{\alpha_{p,m}}{2\pi}|J_{p+1,\nu=1}\rangle\right|} \tag{6.67}$$

$$\Phi_n^{(J)} = \frac{V\langle J_{p,n}|\Psi\rangle}{\left|\langle \frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle\right|} \tag{6.68}$$

$$\mathbb{T}_{\mathrm{m,n}} = \frac{2\langle r_m|J_{p,n}\rangle}{2\pi RV\left|\langle \frac{\alpha_{p,n}}{2\pi}|J_{p+1,\nu=1}\rangle\right|\left|\langle \frac{\alpha_{p,m}}{2\pi}|J_{p+1,\nu=1}\rangle\right|} \tag{6.69}$$

and the two transformations can be written as matrix multiplication:

$$\Phi^{(r)} = \mathbb{T}_{\mathrm{m,n}}\Phi^{(J)}$$
$$\Phi^{(J)} = \mathbb{T}_{\mathrm{n,m}}\Phi^{(r)}$$

6.6.4 The Propagator

Once the radial electric field distribution is known in terms of it's Bessel Function components, a method referred to as the Split-Step Method is used to solve the Wave Equation. Equation 6.15 is split into two equations,

$$\begin{aligned}
\frac{\partial^2\Psi}{\partial z^2} - 2ik_0n_0\frac{\partial\Psi}{\partial z}\frac{\partial^2\Psi}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi}{\partial r} + k_0^2\left[n^2(r,z) - n_0^2\right]\Psi &= 0 \\
&\downarrow \\
\frac{\partial^2\Psi}{\partial z^2} - 2ik_0n_0\frac{\partial\Psi}{\partial z} + \frac{\partial^2\Psi}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi}{\partial r} &= 0 \qquad (6.70) \\
\frac{\partial^2\Psi}{\partial z^2} - 2ik_0n_0\frac{\partial\Psi}{\partial z} + k_0^2\left[n^2(r,z) - n_0^2\right]\Psi &= 0 \qquad (6.71) \\
& \qquad (6.72)
\end{aligned}$$

and the two are solved separately. Notice that the equation 6.70 is just the Wave Equation in a homogeneous media, and equation 6.71 contains the inhomogeneous refractive index terms. The strategy will be to solve equation 6.70, and step forward in the z direction by some Δz. Equation 6.71 can be interpreted as a phase correction, and will be used to adjust the phase of the wavefront, based on the refractive index profile it passed through during the Δz step. Expanding the radial electric field distribution in terms of Bessel Functions, equation 6.70 can be rewritten,

$$\begin{aligned}
\left[\frac{\partial^2}{\partial z^2} - 2ik_0n_0\frac{\partial}{\partial z}\right]\Psi(z)\Psi(r) + \left[\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r}\right]\Psi(z)\Psi(r) &= 0 \\
\left[\frac{\partial^2}{\partial z^2} - 2ik_0n_0\frac{\partial}{\partial z}\right]\Psi(z)\sum_{k=1}^{\infty}a_kJ_p\left(\frac{\alpha_{p,k}}{R}r\right) + \left[\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r}\right]\Psi(z)\sum_{k=1}^{\infty}a_kJ_p\left(\frac{\alpha_{p,k}}{R}r\right) &= 0
\end{aligned}$$

With a little manipulation, the r differential operator on the right-hand side can be replaced by eigenvalues.

$$\begin{aligned}
\left[\frac{\partial^2}{\partial z^2} - 2ik_0n_0\frac{\partial}{\partial z}\right]\Psi(z)\sum_{k=1}^{\infty}a_kJ_p\left(\frac{\alpha_{p,k}}{R}r\right) + \Psi(z)\sum_{k=1}^{\infty}a_k\left[\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r}\right]J_p\left(\frac{\alpha_{p,k}}{R}r\right) &= 0 \\
\sum_{k=1}^{\infty}\left[\frac{\partial^2}{\partial z^2} - 2ik_0n_0\frac{\partial}{\partial z}\right]\Psi(z)a_kJ_p\left(\frac{\alpha_{p,k}}{R}r\right) - \sum_{k=1}^{\infty}\Psi(z)a_k\left(\frac{\alpha_{p,k}}{R}\right)^2J_p\left(\frac{\alpha_{p,k}}{R}r\right) &= 0
\end{aligned}$$

The Bessel Functions are linearly independent, so each index i must separately equal zero,

$$\left[\frac{\partial^2}{\partial z^2} - 2ik_0n_0\frac{\partial}{\partial z}\right]\Psi(z)a_kJ_p\left(\frac{\alpha_{p,k}}{R}r\right) - \Psi(z)\left(\frac{\alpha_{p,k}}{R}\right)^2a_kJ_p\left(\frac{\alpha_{p,k}}{R}r\right) = 0 \tag{6.73}$$

Now Equations 6.70 and 6.73 have the same form. In general, the solution to these differential equations is:

$$\left[\frac{\partial^2}{\partial z^2} - 2ik_0 n_0 \frac{\partial}{\partial z}\right] f(z)\, g(r) - \kappa^2 f(z)\, g(r) = 0$$

$$f(z)\, g(r) = f(z_0)\, g(r)\, e^{-ik_0 n_0} e^{iz\sqrt{(k_0 n_0)^2 - \kappa^2}}$$

If the function is known at some z coordinate, the propagator gives the function at all other z coordinates. Applying this to Equations 6.70 and 6.73, an incident beam profile can be propagated a distance Δz into the medium.

$$\Psi'(\Delta z)\,\Psi'(r) = \Psi(0)\sum_{k=1}^{\infty} a_k J_p\left(\frac{\alpha_{p,k}}{R} r\right) e^{-ik_0 n_0} e^{i\Delta z\sqrt{(k_0 n_0)^2 - \left(\frac{\alpha_{p,k}}{R}\right)^2}} \tag{6.74}$$

$$\Psi(\Delta z)\,\Psi(r) = \Psi'(0)\,\Psi'(r)\, e^{-ik_0 n_0} e^{i\Delta z k_0 n(r,z)} \tag{6.75}$$

First the wavefront is propagated a distance Δz with equation 6.74, then Equation 6.75 is used to adjust the wavefront phase to account for the refractive index profile in the slice Δz.
In terms of the vectors defined in 6.67 and 6.68, this can be written:

$$\Phi_n^{(J)}(z + \Delta z) = \Phi_n^{(J)}(z)\, e^{-ik_0 n_0} e^{i\Delta z\sqrt{(k_0 n_0)^2 - \left(\frac{\alpha_{p,n}}{R}\right)^2}}$$

$$\Phi_m^{(r)}(z + \Delta z) = \Phi_m^{(r)}(z)\, e^{-ik_0 n_0} e^{i\Delta z k_0 n(r,z)}$$

Chapter 7

Monte Carlo Scattering

Source terms within highly scattering media are computed using a Monte Carlo simulation. The Monte Carlo approach to photon transport uses a statistical method to compute the photon distribution within a tissue that may be scattering the photons. The implementation in the BTEC model utilizes work by Wang and Jacques [12].

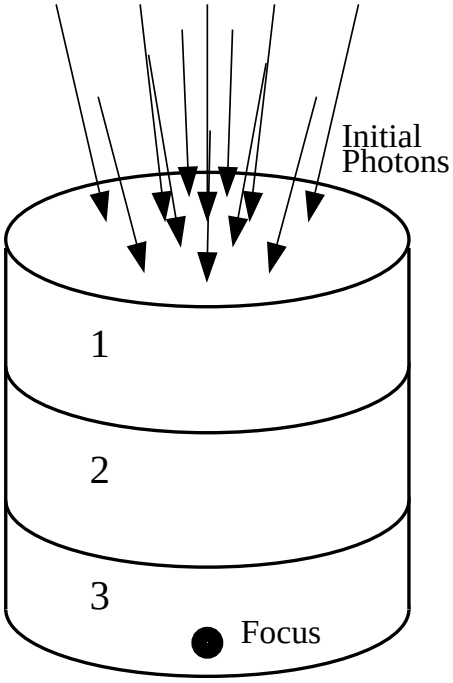

Figure 7.1: Photon Launch

7.1 Theory

As mentioned, the Monte Carlo method for modeling photon transport is based on statistical distributions and random numbers. Photons are essentially modeled as ballistic objects. A packet, or bundle, of photons with weight, w, is launched into a tissue and propagated like ping-pong balls, randomly interacting with the tissue.
Photon propagation is done on a 3-dimensional, Cartesian coordinate system. A photon packet has a position vector, given by the coordinate vector, **r**, in the global coordinate system, and a unit direction vector, **n**, that specifies the

direction the packet is currently traveling. The packet propagates a distance, s, along the direction vector from $\mathbf{r_i}$ to $\mathbf{r_f}$, where $\mathbf{r_f}$ is given by

$$\mathbf{r_f} = \mathbf{r_i} + \mathbf{sn} \ . \tag{7.1}$$

After each propagation, the packet interacts with the tissue, where some of the photons are absorbed, corresponding to a decrease in the packet weight, and the rest are scattered. The packet is propagated in the new direction and the process is repeated until all photons in the packet are absorbed.

The photon absorption is recorded on a 2-dimensional cylindrical grid. At each interaction site, the closest grid point is determined from the $r = \sqrt{x^2 + y^2}$ and z coordinates of the photon packet, and the absorbed packet weight is tallied at the grid location. If the packet is determined to be outside the boundaries of the grid, then it is killed, and a new packet is launched. Any remaining weight is added to one of three tallies, indicating which external boundary the packet crossed (reflected, transmitted, or lateral).

As more photon packets are launched into the tissue, a photon weight distribution is compiled on the grid, representing the number of photons absorbed at (or near) each grid point in the computational space. The distribution is initially very discontinuous, but smooths out as more photon packets are propagated. Once photon propagation is complete, this distribution is transformed into a photon density distribution, differing from the photon weight distribution because of the cylindrical coordinates of the grid. The photon density distribution is normalized to the power of the simulated emitter (minus the percentage of photon weight that was not absorbed on the grid). The resulting distribution gives the source term for the heat equation.

The amount of computation required for the Monte Carlo simulation varies greatly with the model configuration. As a general rule though, the amount of time a simulation requires increases with higher scattering coefficients, and lower absorption coefficients. This happens to be the two instances in which the model is needed most, so the simulation time is inherently long. Therefore, photon propagation is only computed when necessary. The Monte Carlo routine only runs when the emitter is actually turned on in the thermal simulation. After the source term is computed, it is saved, and reused for future time steps in the heat model. This method is valid as long as there are no changes in the thermal model that will affect the propagation simulation (such as temperature dependent absorption and scattering coefficients).

7.1.1 Photon Step Size

The distance a photon travels between interaction sites is determined by sampling the probability distribution of the photon's free path, s. After each tissue interaction, the photons free-flight step size is determined using the interaction coefficient [12], μ_t,

$$s = \frac{-ln(\xi)}{\mu_t} \tag{7.2}$$

where ξ is a random number between 0 and 1 that is uniformly distributed. The interaction coefficient gives the probability interaction per unit length, and is related to the absorption and scattering coefficients of the tissue:

$$\mu_t = \frac{1}{\mu_a + \mu_s} \tag{7.3}$$

Note that the term $-ln(\xi)$ can be interpreted as a dimensionless free-flight distance. The actual step size is computed by dividing this dimensionless quantity by the interaction coefficient of the tissue. If the photon propagates across multiple tissues, the dimensionless step size must be scaled by the interaction coefficients of each tissue propagated through. The actual free-flight distances through each tissue are related to the dimensionless step size by

$$\sum_i \mu_{ti} s_i = -ln(\xi) \ , \tag{7.4}$$

where μ_{ti} is the interaction coefficient of the ith tissue, and s_i is the distance traveled in that tissue. Boundary interactions will be discussed in section 7.1.3

7.1.2 Tissue Interaction: Absorption and Scattering

After a step size is chosen, the photon packet is propagated that distance along the direction vector. At the new position, part of the packet's weight is absorbed. The absorbed weight is computed as

$$dw = \frac{\mu_a}{\mu_t} w \, . \tag{7.5}$$

The weight of the photon packet is decreased by this amount, while the total weight at the nearest grid point is increased.

$$w = w - dw \tag{7.6}$$

$$A[i][j] = A[i][j] + dw \tag{7.7}$$

Here A has been used to denote the array corresponding to the cylindrical grid storing the photon weight distribution. Next the packet is scattered, which just corresponds to a change in the packet's direction unit vector. Scattering involves two angles: a deflection angle, $\Theta \rightarrow [0, \pi]$, and a rotation angle, $\Phi \rightarrow [0, 2\pi]$. The anisotropy of the tissue, g,

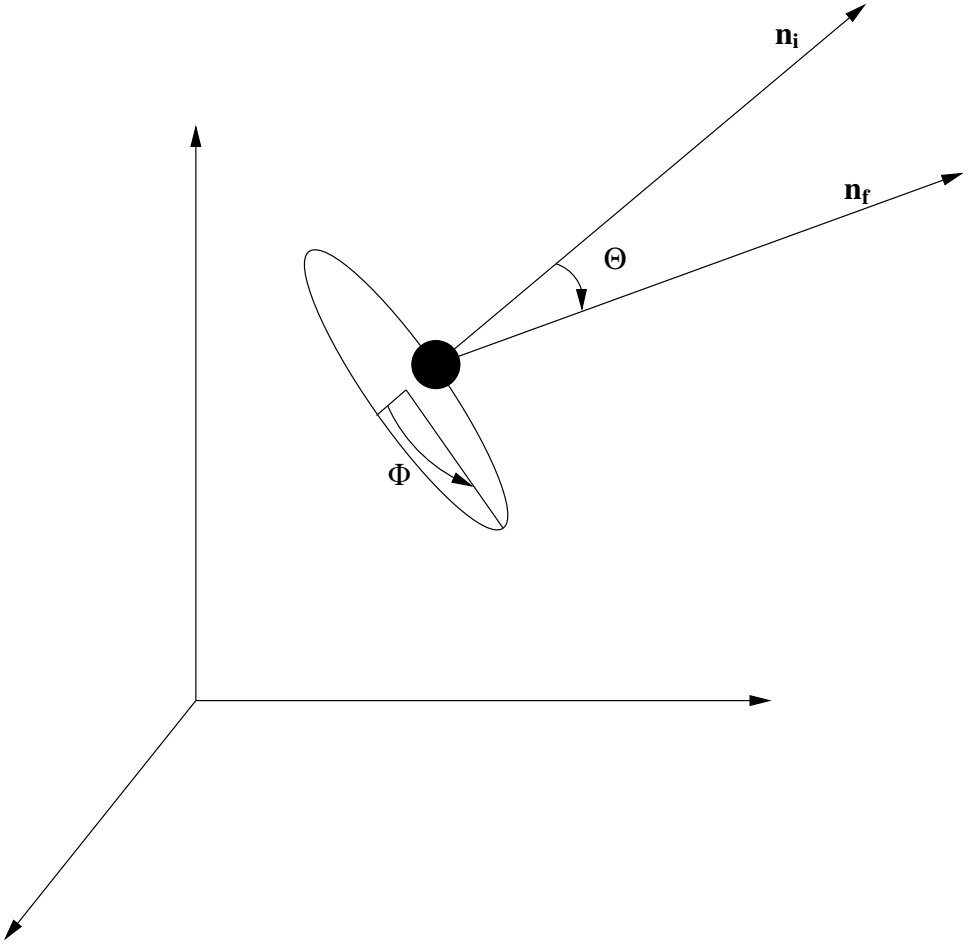

Figure 7.2: Photon's direction **n** is deflected by an angle Θ from the propagation axis, and rotated about the axis by Φ. (Photon not to scale)

is a measure of the expected value of $cos\Theta$. The probability distribution of $cos\Theta$ is given by

$$P\,(cos\ \Theta) = \frac{1 - g^2}{2(1 + g^2 - 2\ g\ cos\ \Theta)^{3/2}} \, , \tag{7.8}$$

so for each scattering event, $cos\Theta$ can be randomly chosen using a uniformly distributed random number, ξ, between 0 and 1:

$$cos\,\Theta = \begin{cases} \frac{1}{2g}\left\{1 + g^2 - \left[\frac{1-g^2}{1-g+2g\xi}\right]^2\right\} & if\ g \neq 0 \\ 2\xi - 1 & if\ g = 0 \end{cases} \tag{7.9}$$

The second angle, Φ, has a uniform probability of being between 0 and 2π. The photon is just as likely to scatter left, as it is to scatter right:

$$\psi = 2\pi\xi \tag{7.10}$$

A new step size is chosen, and the packet is propagated along the new direction.

7.1.3 Boundary Interaction: Reflection and Refraction

The photon packet has a possibility of intersecting a tissue boundary during a propagation step. If so, the packet should not be propagated across the boundary transparently for two reasons. One being that the step size chosen is only correct for the tissue at the packet's initial position. If the packet propagates into another tissue, the step size must be rescaled for the new tissue. Secondly, if the refractive index of the two tissues differs, the packet will either be reflected off the boundary, or refracted across the boundary.
The probability that the packet is reflected at a given incident angle is given by the reflection coefficient:

$$R(\theta_i) = \frac{1}{2}\left[\frac{sin^2(\theta_i - \theta_t)}{sin^2(\theta_i + \theta_t)} + \frac{tan^2(\theta_i - \theta_t)}{tan^2(\theta_i + \theta_t)}\right] \tag{7.11}$$

When a packet reaches a boundary, a random number, ξ, uniformly distributed between 0 and 1, is chosen. The incident angle, θ_i, is then computed. If $\xi < R(\theta_i)$, the packet is reflected, otherwise the packet is transmitted.
With tissue layers being perpendicular to the z-axis, reflection simply corresponds to a change in sign of the z-component of the photon's direction vector, and refraction is given by Snell's Law:

$$n_1 sin(\theta_1) = n_2 sin(\theta_2)$$

Care must be taken when θ approaches 0. If the photon travels some distance, d_1, before intersecting the tissue boundary, then the distance left to travel is $t = s_1 - d_1$. If the photon is reflected, propagation continues in the new direction, with the new step size, t. If the photon is refracted across the tissue boundary, the travel distance must be scaled for the new tissue. Initially, a dimensionless step size is chosen, and scaled by the interaction coefficient of the tissue to give an actual step size (equation 7.2). The dimensionless quantity for the distance left to travel is given by:

$$s' = \mu_{t1}(s_1 - d_1)$$

This quantity is scaled by the interaction coefficient of the second tissue to give the new step size:

$$s_2 = \frac{s'}{\mu_{t2}} = \frac{\mu_{t1}}{\mu_{t2}}(s_1 - d_1)$$

After the boundary interaction, propagation continues. If other boundaries are hit, the process is repeated.

7.1.4 Packet Termination

Packets are terminated when their weight drops below a threshold. The threshold is set to be 0.01% of the initial packet weight. However, the packet cannot simply be terminated once the weight drops below threshold. To conserve energy, a method called Russian Roulette [12] is used to terminate the packet. If the packet weight is below threshold after an absorption event, it enters a game of roulette. The packet has a 1 in 10 chance of surviving the game. If it survives, it propagates another step, but will have to play roulette again. If it does not survive the roulette, it is terminated and a new packet is launched.

7.1.5 Calculating the Source Term

After a complete Monte Carlo simulation, we are left with a grid that contains the photon count at each point. This photon count needs to be turned into a power density corresponding to the emitter power. The grid is a 2-dimensional slice from a 3-dimensional cylinder. Therefore, the photon counts at each grid point represent the totals for a ring in the cylinder. At a specific grid point, $A[i][j]$, the volume of the corresponding ring is

$$V = \pi \left[\left(r[i] + \frac{\Delta r_+}{2} \right)^2 - \left(r[i] - \frac{\Delta r_-}{2} \right)^2 \right] \left[\frac{\Delta z_+ + \Delta z_-}{2} \right] .$$

The photon density on the 2-dimensional cylindrical grid is computed from the photon totals as

$$A[i][j] = \frac{A[i][j]}{V[i][j]} ,$$

where $V[i][j]$ is the corresponding volume element for the point $A[i][j]$. $A[i][j]$ contains the photon density, which is directly proportional to the power density. During photon propagation, four quantities are tracked: total weight launched, total weight reflected, total weight transmitted and total weight lost laterally. The ratio,

$$\alpha = \frac{w_t - (w_r + w_t + w_l)}{w_t},$$

is the ratio of power deposited in the tissue to power input. To compute the power density, the photon density is normalized to the amount of power deposited in the tissue by the emitter,

$$\alpha P_e = 2\pi \int_0^{r_1} \int_{z_0}^{z_1} A\,(r, z)\, r^2 dr dz .$$

Integration is taken over the limits of the computational space. After normalization, the grid, $A[i][j]$, contains the necessary source term for the heat equation.

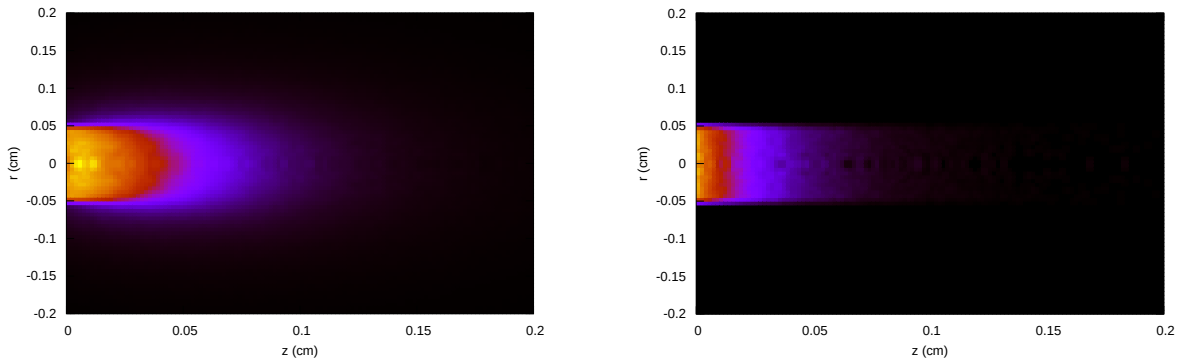

Figure 7.3: Two computed source terms, with (left) and without (right) scattering.

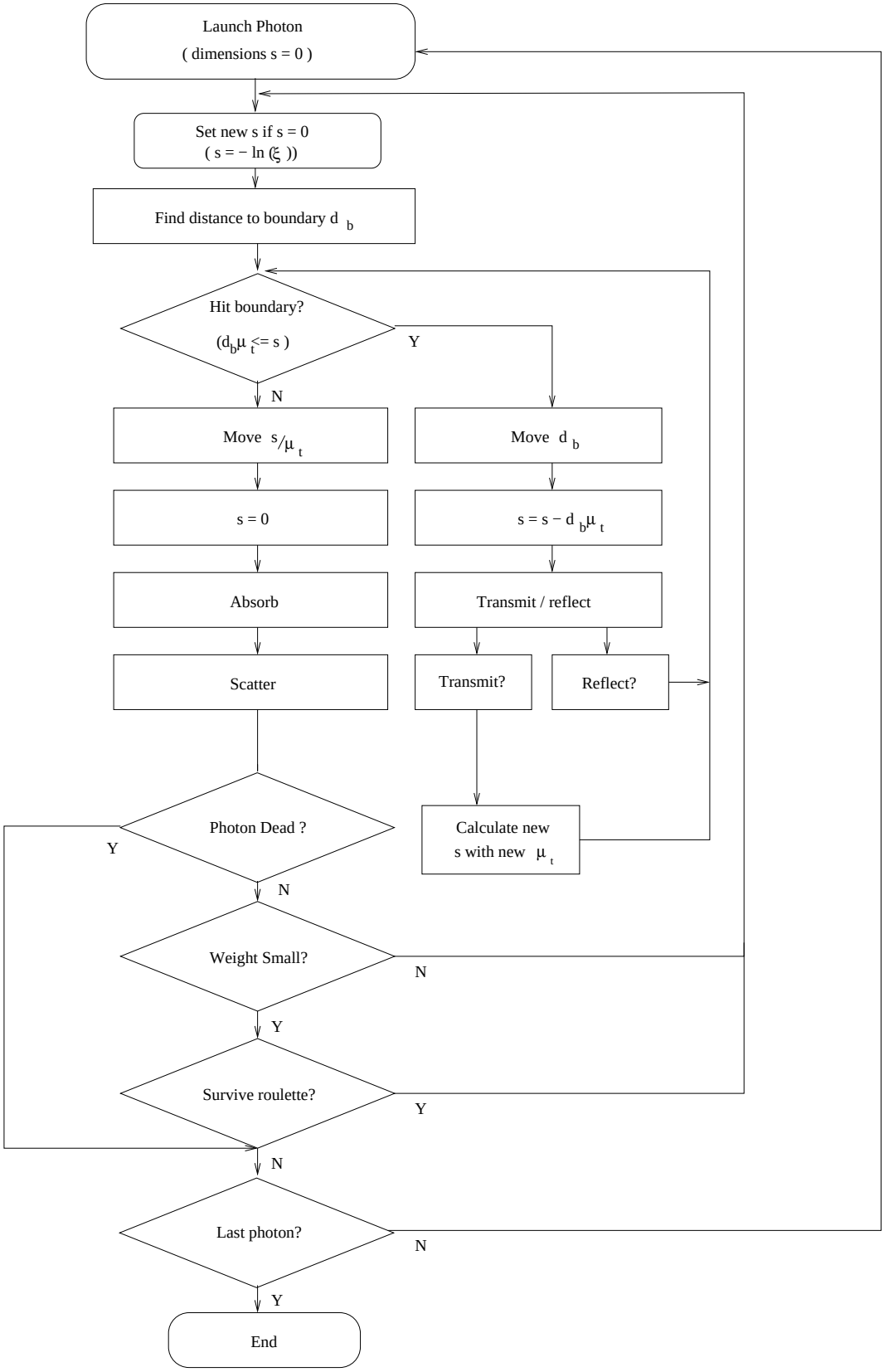

Figure 7.4: Scattering progression and logic

Chapter 8

Z-scan Simulation

8.1 Background and Geometry

A z-scan experiment is commonly used to observe thermal lensing. In a z-scan experiment, a laser is focused and the output is observed through a circular aperture. A cuvette sample is placed at different distances on each side of the focal point. Water will be used for this discussion, although other media may be used. The requirement is that the material's refractive index changes with temperature. As temperature rise in the sample causes the beam to refocus because of the refractive index change, the power transmission through the aperture is recorded. This is done with the sample at many locations between the lens and the aperture along the beam axis, hence "z-scan."

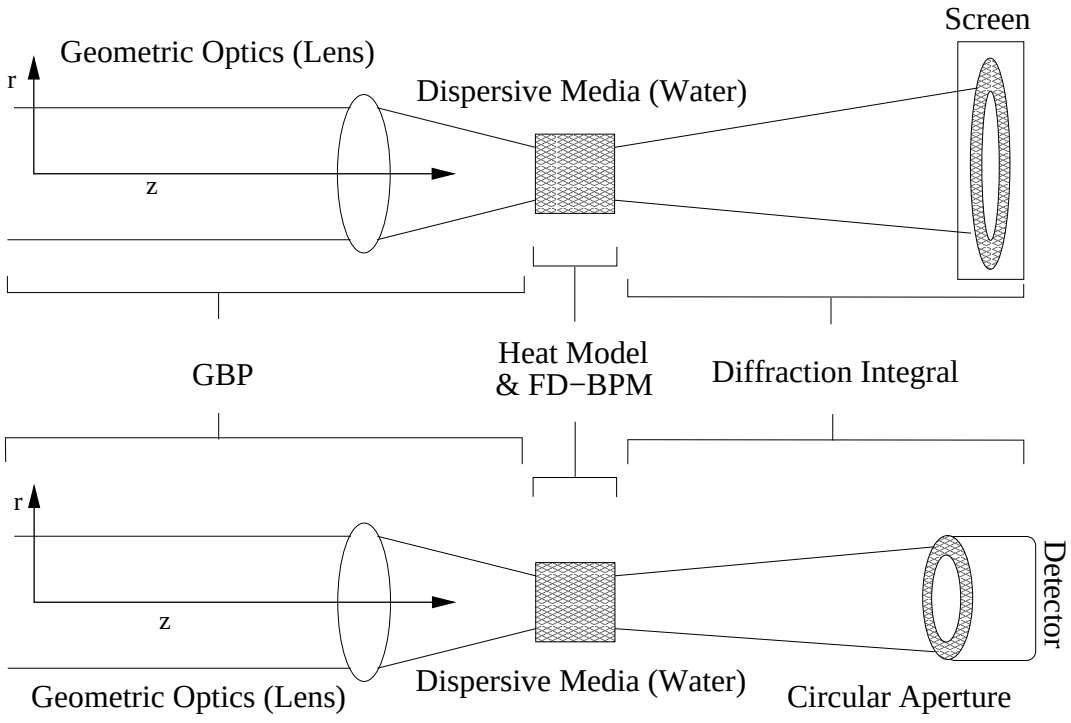

Figure 8.1: Typical problem geometry of a Z-scan experiment

Figure 8.1 is a simplified example of the z-scan experiment. In this figure, the laser beam comes in from the left. It passes through geometric optics and then through the water sample. The beam passes through a fixed circular aperture

and then is either observed on a screen or the power is measured with a detector. Again, the water sample is moved to many positions between the lens and the circular aperture.

The BTEC model is equipped to simulate the z-scan experiment. It uses the Gaussian Beam Propagation model discussed in chapter 5 to calculate the electric field through the linear optics and atmosphere up until the dispersive sample. The finite difference beam propagation (chapter 6) model and the two-dimensional heat model (chapter **??**) are used together to propagate the laser through the sample and compute the temperature rise. To the right of the sample in figure 8.1, the Kirchhoff diffraction integral is used to calculate the electric field intensity at all points. The model can show how the electric field would look projected on a screen (see the top diagram of figure 8.1) and can compute percent transmittance. The diffraction code numerically approximates the percent transmittance based on the electric field distribution at the aperture plane and the aperture diameter defined in the configuration file.

8.2 Derivation of Diffraction Integral

The electric field diffraction pattern on a plane a distance d from from the imaging plane is

$$E(R,\theta) = \frac{i\pi}{\lambda d} e^{-ik_0 R} D(\theta) \tag{8.1}$$

$$R = \sqrt{q^2 + d^2} \tag{8.2}$$

where q is the radial distance on the imaging plane.

Integrate the function over all radial input points and then integrate again over all q on the imaging surface:

$$E = E(r)$$

$$D(\theta) = \int_0^q \frac{q}{R} dq \int_0^{np-1} E J_0\left(\frac{2\pi r}{\lambda}\right) r dr \tag{8.3}$$

where R is not a function of r.

Express in terms of E, noting that it is complex:

$$E(R,\theta)_{imag} = \frac{i\pi}{\lambda d} e^{-ik_0 R} \int_0^q \frac{q}{R} dq \int_0^{np-1} Im(E) J_0\left(\frac{2\pi r}{\lambda}\right) r dr \tag{8.4}$$

$$E(R,\theta)_{real} = \frac{i\pi}{\lambda d} e^{-ik_0 R} \int_0^q \frac{q}{R} dq \int_0^{np-1} Re(E) J_0\left(\frac{2\pi r}{\lambda}\right) r dr \,, \tag{8.5}$$

where $J_0()$ is a first-order Bessel function found by a Numerical Recipes [9] algorithm called BessJo.

In order to get power, E is multiplied by its complex conjugate and integrated over all radial points. The variable, E, here is still a function of the radius and theta.

$$E \times E^* = \left|E(R,\theta)^2_{real}\right| + \left|E(R,\theta)^2_{imag}\right| \tag{8.6}$$

So from the expression:

$$I(R,\theta) = A \int_0^r E^*(R,\theta) E(R,\theta) r_0 dr \,, \tag{8.7}$$

A is a normalization constant with the extra r for the differential in cylindrical coordinates:

$$I(R,\theta)_{tot} = \int_0^{r_{tot}} E^* E r_0 dr = P_T \tag{8.8}$$

where P_T is the total power.

Integrate in parallel over all points up to the limiting aperture:

$$I(R,\theta)_{ap} = \int_0^{r_{ap}} E^* E r_0 dr = P_A \tag{8.9}$$

where P_A is the power with the limiting aperture.

Divide to get fractional intensity:

$$\frac{P_A}{P_T} = D(r_{ap}) \tag{8.10}$$

This is the fractional intensity (W or J) through the measurement aperture.

8.3 Results

A comparison between a z-scan experiment and the BTEC model is shown in figure 8.2.

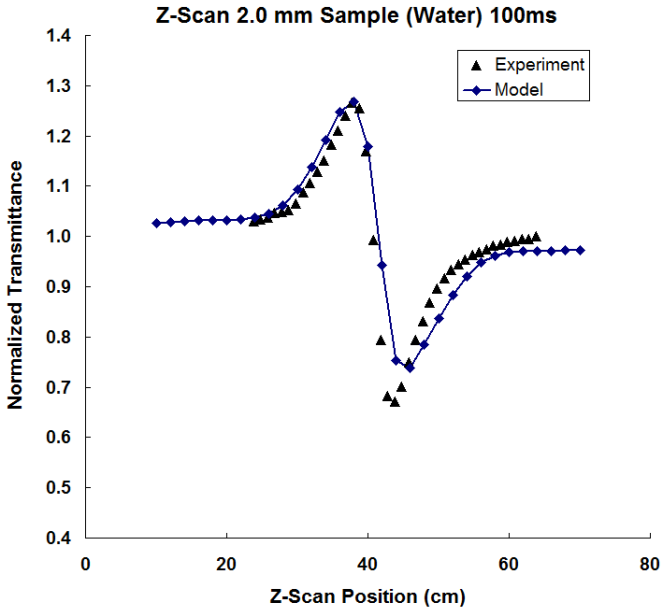

Figure 8.2: Comparison between model and experiment

The experiment was preformed with a 1315-nm laser. The water sample was 0.2 cm thick. This is initial comparisons with some parameter adjustment to estimated material parameters. Each data point was taken with the sample at a different position along the optical axis. At that position the transmittance was recorded at 100 ms.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 9

Damage

9.1 Damage Integral

The current BTEC heat-transfer and thermal injury model produces a solution to the 2-D heat equation that provides an approximate time-dependent solution of the form, $T(z, r, t)$, giving the temperature distribution in cylindrical coordinates at a specific time. At each point within the computational space, the model continually updates a running sum of the Arrhenius integral given below:

$$\Omega(z, r, \tau) = A \int_0^{\tau} exp\left(\frac{-E_a}{RT(z, r, t)}\right) dt \tag{9.1}$$

In the computational space z and r are indexed by (i, j) in a fixed coordinate space such that the values may be indexed as

$$\Omega_{i,j}(\tau) = A \int_0^{\tau} exp\left(\frac{-E_a}{RT_{i,j}(t)}\right) dt\,. \tag{9.2}$$

Currently, this equation is evaluated employing a simple trapezoid method which evaluates the integrand at the current and previous time step, maintaining a running sum of the integral with time. For a given coordinate, the i, j labeling is ignored, yielding

$$\Omega(\tau + \Delta t) = \Omega(\tau) + \frac{1}{2}\left[exp\left(\frac{-E_a}{RT(\tau + \Delta t)}\right) + exp\left(\frac{-E_a}{RT(\tau)}\right)\right]\,. \tag{9.3}$$

This method suffers from errors as the integrand varies rapidly, and the two points of evaluation are weighted equally, essentially linearizing an exponential function. If this function varies such that the equality in equation 9.4 is not a valid approximation, then the integral is not accurate. In a stretched time-step scheme where the temperature is rising rapidly at the end of a pulse, but near the estimated damage threshold, there may be significant error as that time step contains nearly all of the running summed value for equation 9.3:

$$exp\left(\frac{-E_a}{RT(\tau)}\right) \approx 1 + \left(\frac{-E_a}{RT(\tau)}\right)\tau \tag{9.4}$$

Obviously, improved methods are needed to accurately depict short-pulse damage thresholds, where temperature rise is significant during a time step.

9.2 Threshold Search

In the current BTEC heat-transfer and thermal injury model, an optional search algorithm can be used to determine the threshold of an effect. The search algorithm adjusts the power of an emitter until the threshold criterion is met to within

some user-specified tolerance. The optional search can determine the emitter (or combined emitter) power required to reach a user-specified value of damage threshold or a user-specified value of temperature increase. These may be global criteria, or specified for a certain radial extent. For example, the user may choose to evaluate the threshold power required to obtain a damage integral value of 1.0 anywhere within the tissue, or at a specified radius, such that a lesion size may be used as the threshold determining factor.

The current search algorithm is a midpoint-search method, with the initial search range (max, min) specified as a user-defined input. This method continually bisects the search range until the threshold value is reached. In practice, this method requires about 6-12 iterations in which the BTEC Thermal Model is run through the entire simulation time. The bisection method was selected due to the strongly non-linear nature of the damage integral, for which Newton or secant methods were unstable.

While the current search method is reliable, the currently large workload for the model running many high-resolution serial jobs on the AFRL/RHDO clusters is significant. This is creating typical 1-2 day time for a job to reach execution within the queues. Any reduction in search time would be significant and greatly improve turn-around and efficiency.

Chapter 10

Stability and Accuracy

The question of stability is very important when dealing with numerical methods. Much of the discussion of stability only applies if the problem space is one homogeneous material with sink boundary conditions. The cost of having different layers of materials or more complex boundary conditions introduces the possibility of weakening the numerical strength of the method. The most important thing is to make sure that the methods are not pushed so hard that they become unstable. Fortunately, there are some rules that should keep this from happening in most cases.

10.1 Different Types of Errors

Before looking at the method it is important to know about the two main types of errors. The total error will be the sum of these two errors.

Round-off Error

The first type of error is round-off error. This happens because of a computer's inability to exactly store floating point numbers. This type of error is normally very small and does not cause a problem. However, it has been seen to effect the solution if the solution is changing very rapidly. That is, if the maximum value of the solution changes by an order of magnitude in two or three iterations of the method. If the solution becomes chaotic and evolves very rapidly, it is a good idea to check if it was rounding error. The best way of checking this is to change the data type and see if where the solution becomes chaotic changes. A solution that started to look chaotic on the sixth iteration using the double data type should take more iterations to see the chaotic behavior using the long double data type. Reducing the size of the time step should stop the effect of round-off error.

Truncation Error

Truncation error comes from using finite difference in place of derivatives. This type of error is represented using the big "O" notation. In the BTEC code, the space and time finite differences are of order $O(h^2)$ when using a constant grid spacing, h. The one-dimensional model has all the correct finite differences in space to handle stretching the grid and stay $O(h^2)$. Of course, the h gets bigger with stretching so the error goes up. The two- dimensional model does not use a second derivative finite difference that handles a stretched grid so its truncation error will not stay $O(h^2)$ as the grid is stretched.

10.2 Stability of the Thomas Algorithm

Both the methods used in the BTEC code use the Thomas algorithm to solve a tridiagonal system of linear equations. The Thomas algorithm is a non-pivoting elimination method that is very efficient for solving tridiagonal systems. A tridiagonal system is of the form

$$\mathbf{Ax} = \mathbf{b} \tag{10.1}$$

where **A** is a known matrix with three diagonals, **x** is an unknown vector, and **b** is a known vector. The expanded tridiagonal system can be seen in equation 10.2.

$$\begin{bmatrix} b_1 & c_1 & 0 & & \cdots & & 0 \\ a_2 & b_2 & c_2 & & & & \\ 0 & a_3 & b_3 & c_3 & \ddots & & \vdots \\ & & a_4 & b_4 & c_4 & & \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ & & & & a_{N-1} & b_{N-1} & c_{N-1} \\ 0 & & \cdots & & 0 & a_N & b_N \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{N-1} \\ x_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ \vdots \\ d_{N-1} \\ d_N \end{bmatrix} \tag{10.2}$$

There are three requirements on the terms in matrix **A** to keep the method stable.

(i) $a_i < 0, b_i > 0$, and $c_i < 0$

(ii) $b_i > -(a_{i+1} + c_{i-1})$ for $i = 1, 2, \ldots, N$, defining $c_0 = a_{N+1} = 0$

(iii) $b_i > -(a_i + c_i)$ for $i = 1, 2, \ldots, N$, defining $a_1 = c_N = 0$

For a proof of this, please see [10]. It will be shown that these requirements effect boundary conditions and put a limit on the how much properties of materials can change from layer to layer.

10.3 Crank-Nicholson Method

The Crank-Nicholson Method is used for the one-dimensional model in the BTEC code. The Crank-Nicholson Method has an accuracy of $O(\Delta x^2, \Delta t^2)$ in time and space. It is also said to be unconditionally stability [7]. Unconditionally stable means there is no limit on the value of r where

$$r = \frac{\Delta t\, \kappa}{(\Delta x)^2 \rho\, c}. \tag{10.3}$$

The "no limit" means that r can be as large as desired and the solution will not blow up. This does not mean that the solution will be the correct solution or even make physical sense. To insure a meaningful solution, it has been shown that there is a bound on r [10]. This bound is

$$r < \frac{L}{\pi \Delta x} \tag{10.4}$$

where L is the length of the model. This can be restated as a limit on Δt since that is the easiest parameter to change.

$$\Delta t < \frac{L \Delta x \rho\, c}{\kappa \pi} \tag{10.5}$$

10.3.1 Layers

The previous discussion assumes no changes in material properties in the problem space, which will be referred to as layers. Nor does it give much information on the effect of boundary conditions. To find the limits of the method with boundary conditions and layers, the requirements on the Thomas algorithm must be applied to the method.
By applying requirement (i) of the Thomas algorithm to equation 2.25, the following is obtained using equations (2.13-2.15):

$$2\kappa_i > \delta_x \kappa_i \Delta x_{i+} \tag{10.6}$$

$$2\kappa_i > -\delta_x \kappa_i \Delta x_{i-} \tag{10.7}$$

$$\frac{2\rho_i c_i}{\Delta t} > \frac{\delta_x \kappa_i [\Delta x_{i+} - \Delta x_{i-}] - 2\kappa_i}{\Delta x_{i-} \Delta x_{i+}} \, . \tag{10.8}$$

The only way any of these equations will violate the requirements of the Thomas algorithm is if $\delta_x \kappa_i \neq 0$. This can only happen if κ_i is changing, which occurs on the boundary of where two layers meet. Equation 10.8 will be disregarded because either equation 10.6 or equation 10.7 will be violated first. Equation 10.6 and equation 10.7 provide a limit on how much the thermal conductivity can change going from one layer to another. They can be simplified by assuming a constant grid spacing to

$$4\kappa_i > \kappa_{i+1} - \kappa_{i-1} \tag{10.9}$$

and

$$4\kappa_i > \kappa_{i-1} - \kappa_{i+1}. \tag{10.10}$$

This means that, as a rule of thumb, the difference in thermal conductivity between the two layers needs to be at least less than four times the smallest of the two for the Thomas algorithm and thus the Crank-Nicholson Method to be stable. Running the Crank-Nicholson Method shows that it becomes unstable when this rule of thumb is broken.

10.3.2 Boundary Conditions

The requirements for the Thomas algorithm were checked for boundary conditions and it was found that the boundary conditions do not violate the requirements. In addition, according to [10] for derivative boundary conditions the Crank-Nicholson equations are unconditionally stable. For the convective boundary condition a large number of solutions for various values of h_e and time can be found in a transient temperature chart in figure 2-13 of [6]. These solutions can be used to check the error.

10.4 Peaceman-Rachford Method

The Peaceman-Rachford method is unconditionally stable and has an accuracy of $O(\Delta z^2, \Delta r^2, \Delta t^2)$ [8]. The same conditions layer interfaces that were found for the Crank-Nicholson Method can be found for the Peaceman-Rachford.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 11

Verification

11.1 Introduction

In order to verify the BTEC model, several validation cases were considered. Model predictions were compared to known analytical solutions from the text. In some cases, predictions were also compared experimental results, data from text, or with other implementations of the specific portion of the model.

11.2 Source Term Verification

To verify the calculation of the source term used in the 2-D heat model, the BTEC is run for a short time with a flat-top beam. The source term follows Beer's Law. The beam chosen for source term verification is 0.1 cm in diameter has 100 W of power, and a wavelength of 1315-nm. The beam was set to a zero divergence. The beam is propagated through water for 100 μs. Absorption of a 1315-nm beam in water is assumed to be 1.33 cm^{-1}.

The initial intensity is calculated by

$$I_0 = \frac{P}{\pi r^2} \ , \tag{11.1}$$

where P W is the beam power and r cm is the beam radius.

The source term can then be calculated by using Beer's Law and multiplying by the absorption coefficient:

$$A = \mu_a I_0 e^{-\mu_a z} \tag{11.2}$$

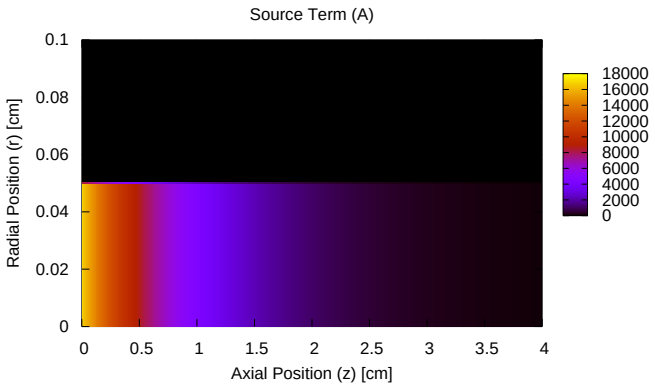

Figure 11.1: Model calculations for a flat-top source term.

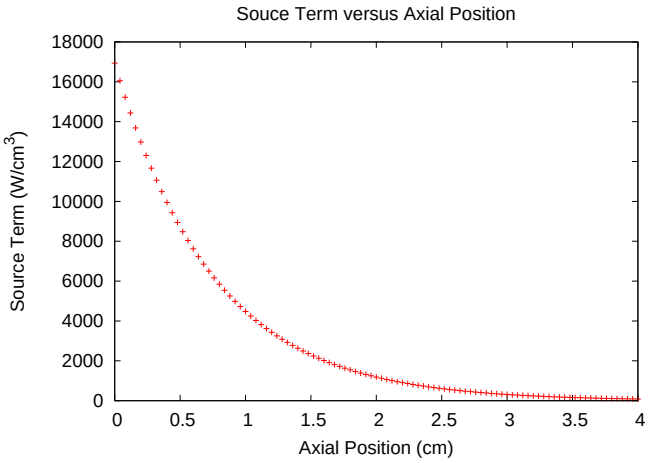

Figure 11.2: Axial distribution of a flat-top source term.

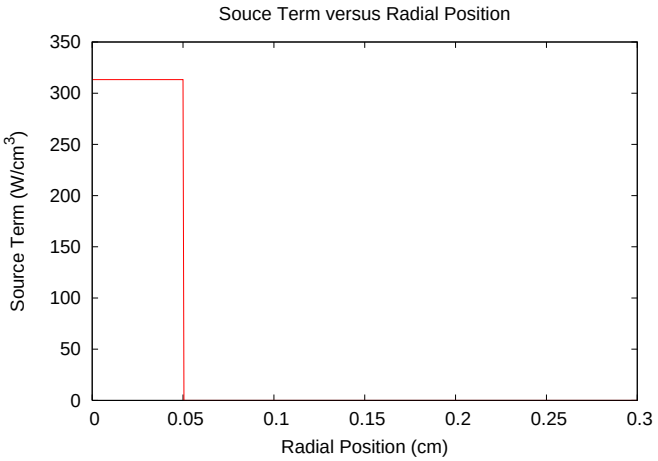

Figure 11.3: Radial distribution of a flat-top source term.

At the front of the sample (z = 0 cm), A is found to be 16934.1 W/cm^3 and the model predicts the same. Further into the sample at z = 3 cm, A is found to be 313.276 W/cm^3 and the model predicts the same.

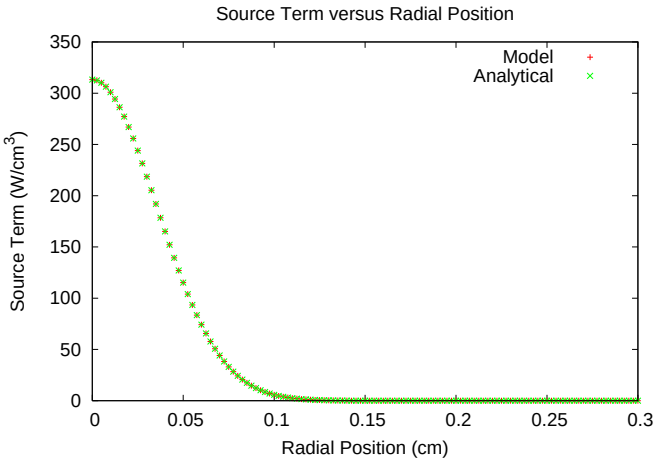

Figure 11.4: Verification of Gaussian source term at an axial position of 3.0 cm

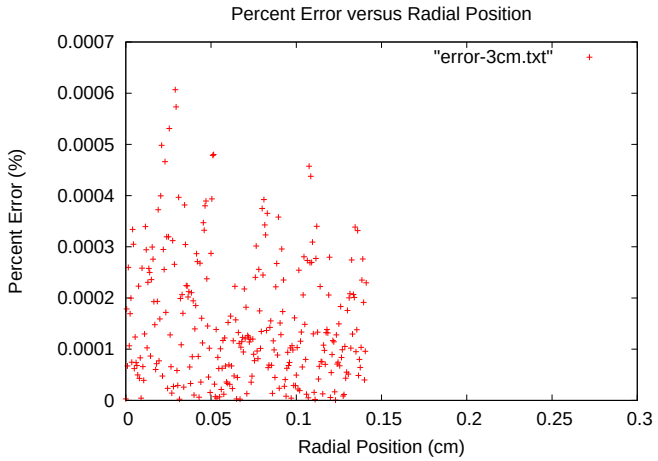

Figure 11.5: Percent error between model and analytical comparisons in figure 11.4

Figure 11.4 shows an analytical solution using Beer's Law to calculate the source term versus the model prediction. The largest percent error between these curves is on the order of 10^{-4} %.

11.3 Thermal Diffusion Verification

To verify the thermal diffusion in the BTEC model, the one-dimensional code is checked against known analytical solutions. Two analytical comparisons are shown below.

11.3.1 One-Dimensional Stretched Grid

The analytical solution in equation 11.3 is from [6]. It is derived for an infinite space ($-\infty < x < \infty$). A constant initial temperature ($T_0 = 100$ °C) is given for the space ($0 < x < L$). The analytical solution was calculated with $L = 1.0$ cm. The BTEC model was run with the constant temperature block from 2.0 cm to 3.0 cm. This is a 2.5 cm coordinate shift from the analytical solution. The BTEC model utilizes a stretched coordinate grid to simulate an infinite problem space. The coordinate shift is necessary to center the constant temperature block of T_0 °C in the center of the uniform grid. The analytical solution is

$$\frac{T(x,t)}{T_0} = \frac{1}{2}\left[erf\left(\frac{L+x}{\sqrt{4\alpha t}}\right) + erf\left(\frac{L-x}{\sqrt{4\alpha t}}\right)\right], \tag{11.3}$$

where again

$$\alpha = \frac{k}{\rho c} \tag{11.4}$$

and

$$\begin{array}{lll} \kappa & = 0.0628 & \left[\frac{W}{cm\ ^\circ C}\right] \\ \rho & = 1.0 & \left[\frac{g}{cm^3}\right] \\ c & = 4.1868 & \left[\frac{J}{g\ ^\circ C}\right]. \end{array}$$

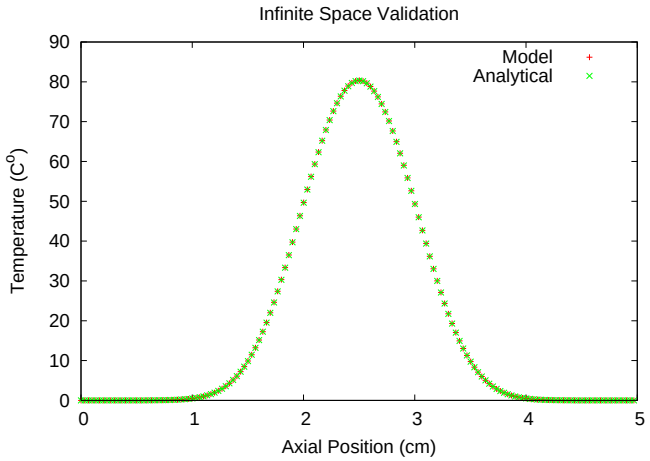

Figure 11.6: Model verifications for simulating an infinite problem space

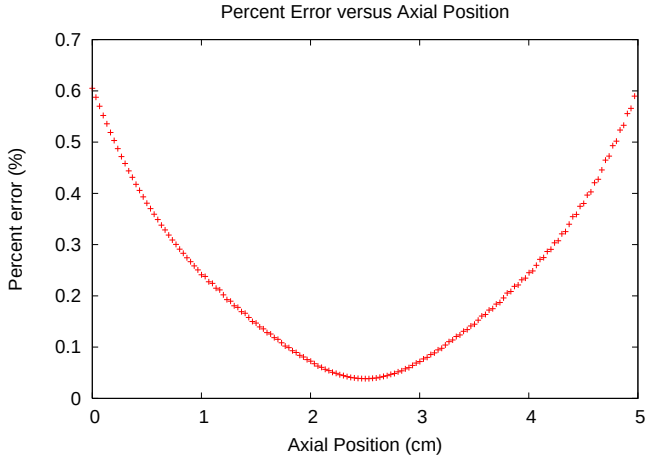

Figure 11.7: Percent error for figure 11.6

11.3.2 One-Dimensional Uniform Grid

In this case, the grid is uniform and extends from 0.0 cm to 5.0 cm. The entire problem space has an initial temperature of 100 °C and the ambient temperature is 0 °C. The left boundary is an insulating boundary and the right boundary is convective. The convective heat transfer rate is $h_e = .2$ W/cm^2°C. The model is compared to an analytical solution form [6] after 5 s. The analytical solution used here is shown in equation 11.5:

$$T(x,t) = 2T_0 \sum_{m=1}^{\infty} e^{-\alpha\beta_m^2 t} \frac{H_2}{L(\beta_m^2 + H_2^2) + H_2} \frac{cos(\beta_m x)}{cos(\beta_m L)} , \tag{11.5}$$

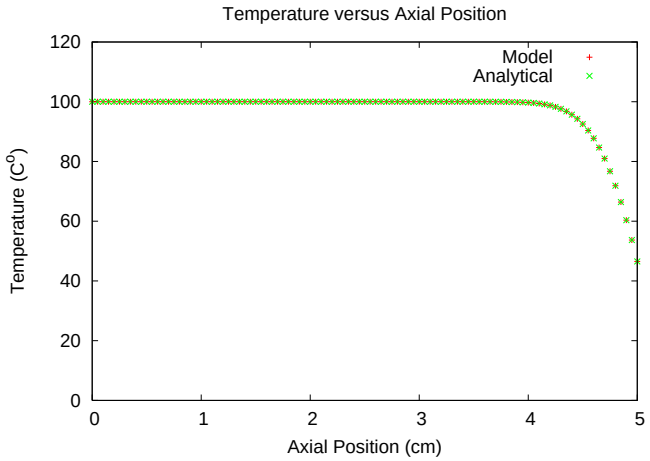

Figure 11.8: Model verifications for convective and insulating boundaries

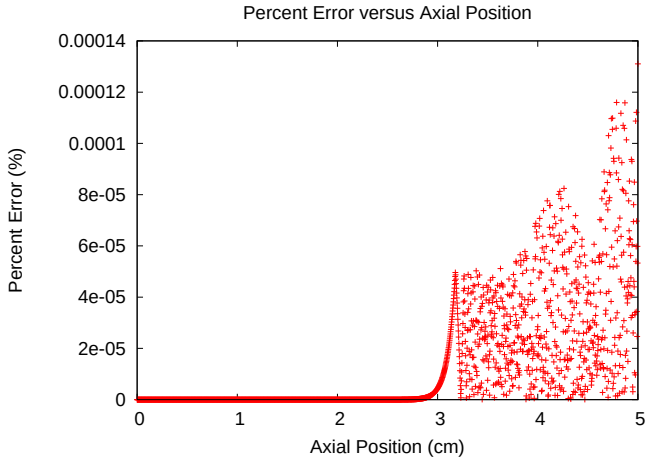

Figure 11.9: Percent error between model and analytical comparisons in figure 11.8

where T_0 °C is the initial temperature of the block of size L cm and H_2 cm^{-1} is a convection term and is defined by

$$H_2 = \frac{h_e}{\kappa} .$$

The β_m terms are the positive roots of

$$\beta_m tan(\beta_m L) = H_2 \tag{11.6}$$

and α term is thermal diffusivity and can be found by

$$\alpha = \frac{k}{\rho c} , \tag{11.7}$$

where

$$
\begin{array}{lll}
h_e & = .2 & \left[\frac{W}{cm^2 °C}\right] \\
\kappa & = 0.0628 & \left[\frac{W}{cm\ °C}\right] \\
\rho & = 1.0 & \left[\frac{g}{cm^3}\right] \\
c & = 4.1868 & \left[\frac{J}{g\ °C}\right] .
\end{array}
$$

The analytical solution 11.5 used for comparison with the BTEC model utilized the first 100 positive roots of equation 11.6

11.3.3 Two-Dimensional Large Beam Verification

The one-dimensional code simulates an infinitely large laser beam. In two-dimensional cases with large beam diameters, the axial temperature distribution of the two-dimensional case should compare to the one-dimensional temperature distribution.

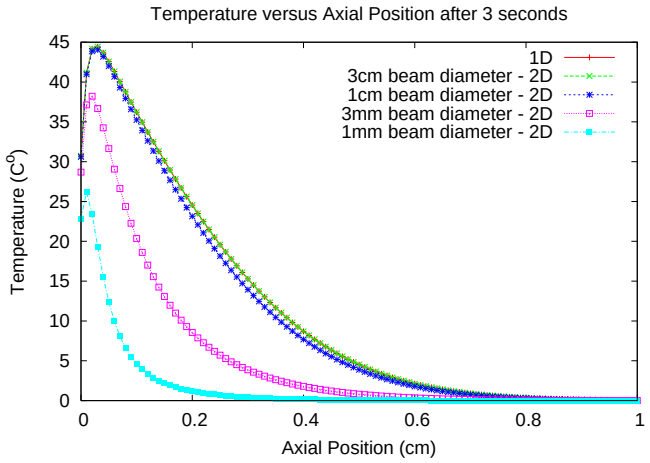

Figure 11.10: Temperature comparison of 1-D and 2-D along z axis

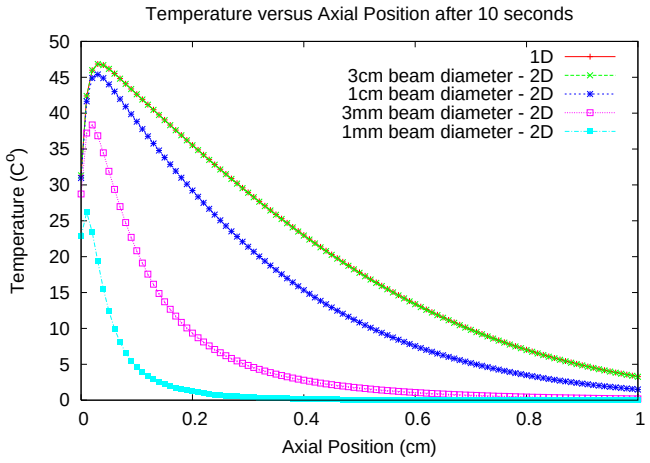

Figure 11.11: Percent error for figure 11.10

Figures 11.10 and 11.11 show how the axial temperatures for four different beam diameters in the two-dimensional model compare with the one-dimensional model. Figure 11.10 shows the temperature distribution after 3 seconds and figure 11.11 shows the temperature distribution after 10 seconds. The power density is kept constant in all runs to $I = 3.537$ W/cm^2. The absorption coefficient is $\mu_a = 100$ cm^{-1}.

11.3.4 Two-Dimensional Mainster Source

The Mainster source found in [3] is a constant cylindrical heat source embedded in the problem space as shown in figure 11.12. The BTEC model has a Mainster source option. When this option is enabled, the power, P W, defined in the emitter configuration file is used point as the source A W/cm^3. The cylinder source uses the beam diameter defined in the emitter configuration file and is only added to the layer defined with its layer type equal to 1. To produce the embedded source, three layers are defined and the center one is defined as layer type 1 from within the layer configuration file.

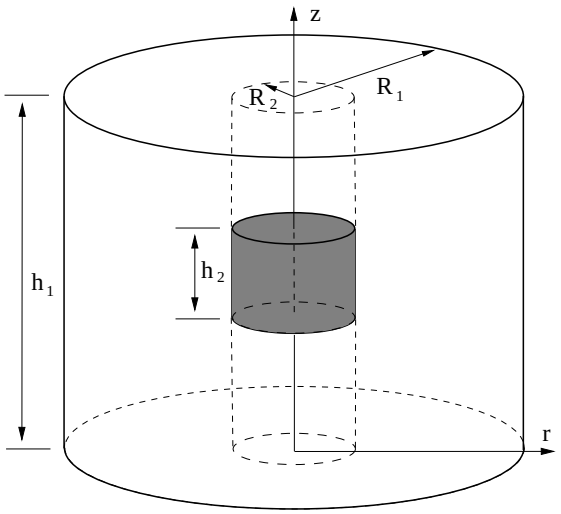

Figure 11.12: Embedded Mainster source

The analytical solution from 11.13 for this problem geometry is shown in equation 11.8:

$$T = \frac{A}{4}\sqrt{\frac{\rho c}{\pi \kappa}} \int_0^t \frac{dt'}{(t-t')^{3/2}} \int_{-0.03}^{0.03} e^{\frac{-(z')^2}{4(t-t')}\frac{\rho c}{\kappa}} dz' \int_0^{0.105} e^{\frac{-(r')^2}{4(t-t')}\frac{\rho c}{\kappa}} r' dr' \tag{11.8}$$

The parameters used in the model for comparison to the analytical solution in 11.8 are as follows:

$$\begin{array}{rll} R_1 & = 1.0 & [cm] \\ h_1 & = 2.0 & [cm] \\ R_2 & = 0.105 & [cm] \\ h_2 & = 0.06 & [^\circ\mathrm{C}] \\ \kappa & = 0.00586 & \left[\frac{W}{cm\ ^\circ\mathrm{C}}\right] \\ \rho & = 1.0 & \left[\frac{g}{cm^3}\right] \\ c & = 4.184 & \left[\frac{J}{g\ ^\circ\mathrm{C}}\right] \\ A & = 41.84 & \left[\frac{W}{cm^3}\right] \\ \mu & = 10.00 & \left[\frac{1}{cm}\right] \end{array}$$

All boundary conditions are considered sinks in the Mainster case. The temperature recorded is the maximum temperature rise and is found at the center of the heat source.

Time [s]	Analytical Temperature [°C]	BTEC Temperature [°C]	Percent Error [%]
0.001	0.01	0.010	0.0
0.002	0.02	0.020	0.0
0.005	0.05	0.050	0.0
0.01	0.09	0.100	0.0003
0.02	0.19	0.200	0.0085
0.05	0.49	0.498	0.0689
0.1	0.97	0.979	0.2012
0.2	1.83	1.830	0.2180
0.5	3.831	3.848	0.4301
1.0	6.116	6.12	0.0601
2.0	8.797	8.90	1.1485
5.0	12.14	12.1	0.3678
10.0	14.15	14.1	0.3674
20.0	15.68	15.7	0.1076
50.0	17.09	17.1	0.0152
100.0	17.82	17.8	0.1348
200.0	18.33	18.3	0.1825
500.0	18.69	18.8	0.5383
1000.0	18.77	19.0	1.1805

Table 11.1: Table of values for figures 11.13 and 11.14.

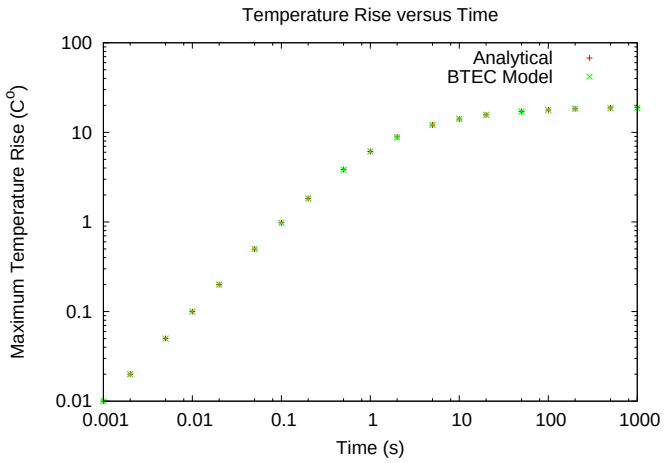

Figure 11.13: Maximum temperature rise comparison of analytical solution and BTEC predictions at different times

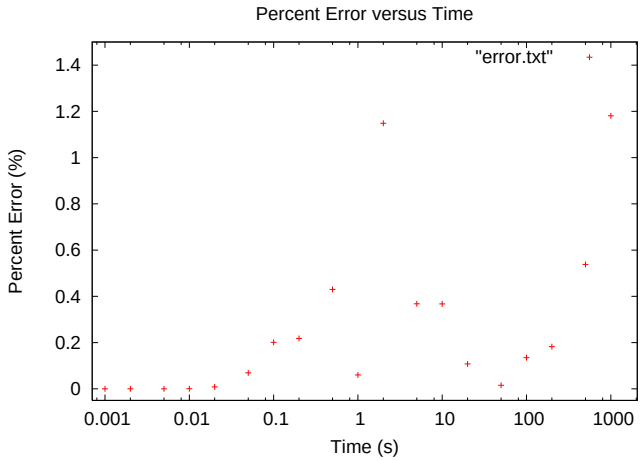

Figure 11.14: Percent error of BTEC as compared to analytical solution with Mainster source term

11.4 COMSOL Thermal Diffusion Comparison

The BTEC model data was compared to COMSOL data provided by Dr. Irwin Goldberg and Misty Garcia. The COMSOL model was a two-dimensional Cartesian model with a circular source in the center. The source was a 5 mm beam that provides 100 W/cm^3 as a source term for the model. A constant source in a two-dimensional model is the same as and infinite source in a three-dimensional model. The BTEC was configured as a constant source from $z = 0$ cm to $z = 10$ cm. Only points in the $z = 5$ cm plane were considered. The maximum radial boundary was simulated to be infinity as well. Several points were compared between the two models. Figure 11.15 shows two points and how they compare between the two models.

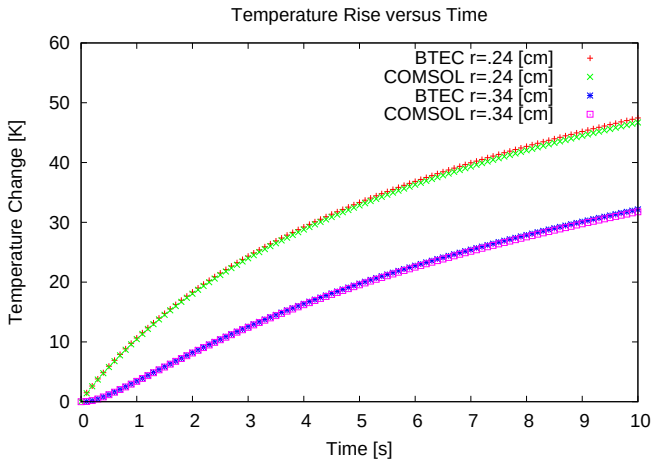

Figure 11.15: Infinite source COMSOL comparison

The models were compared in figure 11.15 at points 2.4 mm and 3.4 mm from the center of the source. The first point would be slightly inside the source and the second is outside the source.

11.5 Damage Integral

To validate the BTEC implementation of the Arrhenius damage integral (equation 9.1), a separate model was created in Matlab. The model used an array of temperature versus time values to numerically compute the value of the damage integral at each time included in the array. Temperature versus time and damage versus time arrays were created for a specific grid point using the BTEC model. The temperature versus time array was used in the Matlab model to create a damage versus time array which is compared to the BTEC's damage versus time array in figure 11.16.

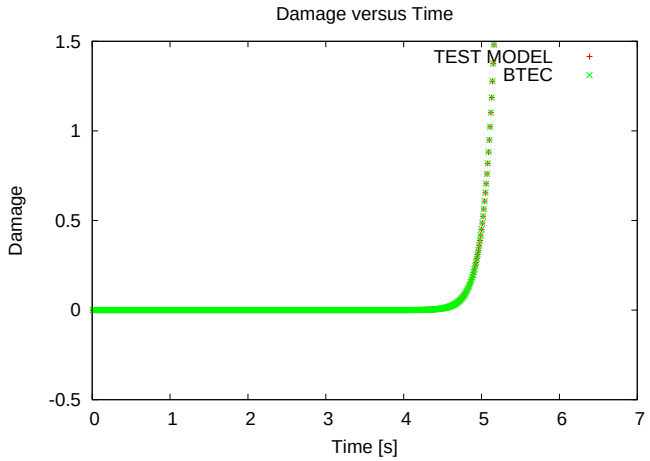

Figure 11.16: Probability of damage comparisons between BTEC and Matlab models

This comparison was done with a constant laser source of 50 W with a beam diameter of 0.1 cm. The material had the following parameters:

$$\begin{array}{lll} \kappa & = 0.0628 & \left[\frac{W}{cm\ ^\circ\mathrm{C}}\right] \\ \rho & = 1.0 & \left[\frac{g}{cm^3}\right] \\ c & = 4.1868 & \left[\frac{J}{g\ ^\circ\mathrm{C}}\right] \\ \mu & = 1.345 & \left[\frac{1}{cm}\right] . \end{array}$$

11.6 Beam Propagation

To help verify that the beam propagation methods in the BTEC model are working correctly, three separate methods are compared for the same case. A Gaussian beam is focused for 1 cm using the GBP, Hankel transform, and Pade beam propagation methods and the results are compared in figure 11.17.

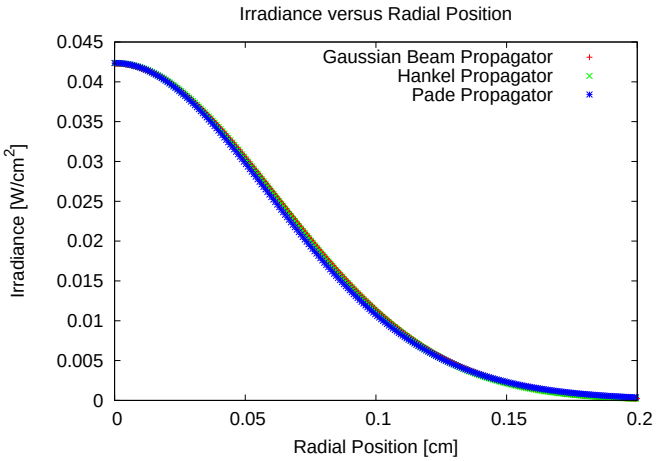

Figure 11.17: Focused Gaussian comparison between Hankel, Pade, and GBP

To accomplish the comparison in figure 11.17, the GBP model is configured with a spherical wave front at $z = 0$ to be propagated by either the Hankel or Pade method into the sample. After initial data are generated (before any heating) with the Pade and Hankel method, data are extrapolated for irradiance at all radial positions for $z = 1$ cm. The model is then configured to have the initialize GBP a plane at $z = 1$ cm. The GBP computes the translation of the Gaussian beam to the $z = 1$ cm plane.

Chapter 12

Configuration and Use

12.1 Overview

The BTEC thermal model is a 2-D (cylindrical coordinate system) simulation of optical-tissue thermal interactions. The code supports the illumination of tissues by optical sources, the temperature response from the linear absorption of optical radiation, and subsequent damage.

An alternating direction implicit (ADI) finite difference method is employed in the solution of the heat equation. The method is commonly referred to as the Peaceman-Rachford method. The simulation can be configured with a source term defined by a single or multiple optical emitters (laser or broadband). The tissue simulation is represented as a one-dimensional stack of homogeneous layers along the z-axis. Each layer can have differing thermal, optical, and physical properties. The linear absorption coefficient of each layer defines the energy transfer from the optical source to the tissue. Boundary conditions along the axial direction may be selected as a sink (temperature held constant) or as a combination of surface boundary conditions (convection, radiative, or evaporative).

The model is configured through three types of configuration files: (1) The main, or top-level configuration file (config.*.in) which specifies the problem geometry, simulation time, boundary condition types, and search parameters used to find tissue damage thresholds. (2) Emitter files (*.emitter) which specify the optical emitter properties such as wavelength or spectrum, power, spatial profile, time-dependent power, and parameters which specify time-step size in the simulation. (3) Layer files (*.layer) which specify thermal, optical, and physical properties of homogeneous tissue layers along with the thickness of each layer.

Running BTECthermal with the options -c will generate template versions of all the configuration files. The templates are listed here.

12.2 Top-Level (config.*.in) Configuration

This is the top-level configuration file for the BTEC Thermal Model. This is the file where the user inputs the desired overall simulation parameters:

The following is a short description of each input:

1. #KeyValue header: The first line of the config must start with the string #KeyValue.
2. Dimensions: (INTEGER, 1 |2) The number of dimensions to use in the simulation. Currently, BTEC supports 1-D, and 2-D cylindrical coordinates.

3. SimulationType: (INTEGER, 0 |2) This option is depreciated and should be set to 0. Previous versions of BTEC separated Propagation emitters from the others. Propagation emitters are now implemented through the Standard Emitter interface, and specified by the StandardEmitter entries below. However, z-scan simulations are still invoked from here.

4. AxialGridType: (INTEGER, 0 |1) The user can choose to use a uniform grid (0), or stretched grid (1) in the axial direction. In a uniform grid, the spacing between values on the axis is constant. The stretched grid adds points to the end(s) of the uniform grid, and spacing between these extra points grows. Grid stretching only occurs at the end of the axis, and will only be used when the boundary condition is set to a SINK for that end. It will not be used if the end of the axis represents a surface boundary. An example of a stretched grid used by Mainster et al. is given on pg 403 of Welch and Gemert's book, *Optical-Thermal Response of Laser-Irradiated Tissue*.

5. Nz: (INTEGER, >1) The number of axial divisions. The axial grid will consist of this number plus one grid points.

6. zMin: (DOUBLE PRECISION, >0.0) The minimum axial coordinate value. Units are centimeters (cm). This coordinate applies to the uniform portion of the grid.

7. zMax: (DOUBLE PRECISION, >0.0) The maximum axial coordinate value. Units are centimeters (cm). This coordinate applies to the uniform portion of the grid.

8. zStretchRatio: (DOUBLE PRECISION, >1.0) The grid spacing stretching ratio for the extended, stretched axial grid. Each Δz calculated by this ratio times the previous Δz. The value must be relatively small for numerical derivative approximations to hold, and to limit the extent of the grid. This parameter is ignored for axial grid type of 0 (uniform grid), and when the minimum or maximum axial coordinate boundary condition is set to a surface boundary condition type.

9. zMinBC: (INTEGER, 0 - 7) The axial grid may be assigned either a sink boundary condition mechanisms). Currently, the model supports the following assignments for the Minimum (and Maximum) Axial Position Boundary Condition. Specific parameters for the boundary conditions are specified within the layer configuration files.

TYPE	VALUE
SINK	0
CONVECTIVE	1
RADIATIVE	2
CONVECTIVE + RADIATIVE	3
EVAPORATIVE	4
CONVECTIVE + EVAPORATIVE	5
RADIATIVE + EVAPORATIVE	6
RADIATIVE + EVAPORATIVE + CONVECTIVE	7

10. zMaxBC: (INTEGER, 0 - 7) See parameter 9 for a description, the same condition types apply.

11. RadialGridType: (INTEGER, 0 |1) If the user selects 2 dimensions, then the second axis (r) can be stretched at the maximum end. Surface boundary conditions are not implemented at the r_{max} boundary, so stretching can always be used when specified here.

12. Nr: (INTEGER, >1) The number of radial divisions. The radial grid will consist of this number plus one grid points.

13. rMax: (DOUBLE PRECISION, >0.0) The maximum radial coordinate, units are centimeters (cm). This coordinate applies to the uniform portion of the grid. If a stretched grid is selected then additional grid points are added to the grid with incremental spacing ratios as set below. (*Note:* The minimum radial coordinate is assumed to be $r = 0$.)

14. rStretchRatio: (DOUBLE PRECISION, >1.0) The grid spacing stretching ratio for the extended, stretched radial grid. Each grid point is spaced by this ratio times the spacing of the previous grid point spacing. The value must be relatively small for numerical derivative approximations to hold, and to limit the radial extent of the grid. This parameter is ignored for radial grid type of 0 (uniform grid).

15. rMaxBC: (INTEGER, 0 |1) The radial grid may be assigned either a sink boundary condition ($\Delta T(r_{max}) = 0$) or an insulating boundary condition $\left(\frac{\partial T}{\partial r}\Big|_{r=r_{max}} = 0 \right)$.

16. TotalSimTime: (DOUBLE PRECISION, >0.0) The simulation time starts at $t = 0.0$ and runs to this user-specified value, given in seconds.

17. dt: (DOUBLE PRECISION, >0.0) The simulation will make time steps of this value, unless an adaptive time step is specified in the first emitter configuration file. In the case of an adaptive time step, this parameter is ignored. Ideally, the time step should be set to a value much shorter than the thermal relaxation time, such that the numerical estimates for derivatives remain accurate.

18. dtMax: (DOUBLE PRECISION, >0.0) In the case of adaptive time steps, the maximum time step is currently not bounded by an estimate for stability. This parameter allows the user to limit the time step to a maximum value if conditions of instability or inaccuracy are observed the solution.

19. TissueBaselineTemp: (DOUBLE PRECISION, <0.0) This is the temperature, in °C, of the tissue corresponding to ΔT equal to zero in the heat equation solution. The value is typically 37.0 for body temperature.

20. AmbientTemp: (DOUBLE PRECISION, <0.0) This value is the temperature of the surrounding atmosphere for conditions where surface boundary conditions are employed. The value must be present but will not be used if both the zmin and zmax boundary conditions are set to SINK.

21. RelHumidity: (DOUBLE PRECISION, 0.0 - 1.0) This value is the relative humidity value for the atmosphere surrounding the sample and is employed only for surface boundary conditions using the evaporative boundary condition type, or combinations of surface boundary conditions using the evaporative boundary condition type.

22. LogDataFlag: (INTEGER, 0 |1) This turns logging on and off. Several files are generated when this flag is set to 1, such as the thermal distribution at some time, the source term at some time, and so on.

23. LogInterval: (INTEGER, >0) This integer specifies the number of time steps between logging. Note that in the case of the non-uniform time step the logged data will not be equally spaced in time.

24. UserDamageThreshold: (DOUBLE PRECISION, >0.0) The valued used in the evaluation of the damage integral to indicated that tissue has been damaged. The damage integral is computed at each grid point during the simulation. If the damage value becomes greater than this value, then the tissue is flagged as damaged.

25. DamageThresholdSearchFlag: (INTEGER, 0 |1) This sets the option for the simulation to search for a threshold power to give a user-specified damage condition. If the user would rather find the damage given by a certain set power and not do a search, this would be set to 0. Entering 1 indicates that a search should be done.

26. MinPowerRatio/MaxPowerRatio: (DOUBLE PRECISION, >0.0) These inputs are the multipliers of the power specified for the emitter. They give the simulation bounds with which to start the search. If the threshold power is outside of this range, the simulation will print an error message and exit.

27. ThreshSearchType: (INTEGER, 0 - 3) There are four different search types. A search type of 0 means search for the threshold to cause damage at any point within the computational grid. A search type of 1 means to look for a specific temperature rise at any point within the grid. A search type of 2 means to look for damage of a given radius. A search type of 3 means to look for a given temperature rise out to a given radius. For cases where damage is being examined, the criterion for damage is that the damage integral is greater than or equal to 1.

28. PrimeThreshValue: (DOUBLE PRECISION, <0.0) This value depends on the search type. For a search type of 0, this parameter is not used. For a search type of 1 or 3, this is the temperature rise. For a search type of 2, this is the radius of damage the must be present as a threshold criteria. (*Note:* At the current model ability damage, area should be equal or greater to the beam area to prevent variations in thresholds with a change in the number of grid points.)

29. SecThreshValue: (DOUBLE PRECISION, >0.0) This number is only used for search type 3. It is the radius at which the temperature rise specified must occur.

30. ConvergThresh: (DOUBLE PRECISION, >0.0) This is the percent error in threshold criterion that the user will allow in their results. The code will iterate until the predicted threshold criterion is within this ratio of agreement, and stops the search.

31. Emitter[i]: (STRING, $i = 0, 1, 2, ...$) The string specifies the path name (relative or full) to the Emitter files. The index i must start at 0, and increases for each Emitter.

32. StandardEmitter[i]: (STRING, $i = 0, 1, 2, ...$) The string specifies the path name (relative or full) to the StandardEmitter files. Propagation emitters and Monte Carlo Scattering emitters are implemented through this interface. The index i must start at 0, and increases for each Emitter.

33. Layer[i]: (STRING, $i = 0, 1, 2, ...$) The string specifies the path name (relative or full) to the Layer files. The index i must start at 0, and increases for each Layer.

34. InitialConditionsFlag: (INTEGER, 0 |1) This flag indicates that the user has specified the loading of an initial temperature distribution within the solution grid. The initial condition is specified by the file below.

35. InitialConditionsFile: (STRING) The file containing the initial conditions temperature profile.

36. PropagationType: (INT) This configuration is obsolete. See StandardEmitter configuration for Propagation emitter configuration.

37. NumPropSteps: (INT) This configuration is obsolete. See StandardEmitter configuration for Propagation emitter configuration.

38. ApertureCoordinate: (DOUBLE PRECISION) This specifies the z coordinate of the aperture for z-scan simulations.

39. ApertureRadius: (DOUBLE PRECISION, >0) The aperture radius for z-scan simulations.

40. NDiffPattern: (INT, >0) The number of points to sample in the diffraction pattern calculations.

41. DiffPatternMaxRadius: (DOUBLE PRECISION, >0.0) Maximum radial coordinate in diffraction pattern calculations.

42. ZScanStartCoord: (DOUBLE PRECISION, <ApertureCoordinate) The initial sample position for z-scan simulation

43. ZScanEndCoord: (DOUBLE PRECISION, <ApertureCoordinate) The final sample position for z-scan simulation

44. ZScanStepSize: (DOUBLE PRECISION, <ApertureCoordinate) Step size taken between initial and final sample positions in z-scan experiment.

45. ZScanLensCoord: (DOUBLE PRECISION, <ZScanStartCoord) Lens coordinate for z-scan simulations. Lens must be placed in front of all sample positions.

46. ZScanLensFocalLength: (DOUBLE PRECISION) The focal length of the lens for z-scan simulations.

12.3 Emitter (*.emitter) Configuration

1. EmitterType: (INTEGER, 1 |2) Set to 1 for laser source, 2 for broadband source.
2. PulseType: (INTEGER, 1 - 3) This determines of the emitter produces a single (1), square pulse ; multiple (2) square pulses; or a user defined (3) pulse train.
3. FocusType: (INTEGER, 1 |2) Specifies if the focus is stationary (1) at some z position, or moves (2).
4. TimeStepType: (INTEGER, 1 |2) The time step can either be fixed (1), with a constant dt, or adaptive (2), where dt grows for each time step[1]
5. ProfileType: (INTEGER, 1 |2) The beam profile can be set to a Gaussian (1), Flattop (2), Annular (3), or User defined (4).
6. PeakPower: (DOUBLE PRECISION, >= 0.0) The peak power of the emitter in W.
7. MinWavelength: (DOUBLE PRECISION, >= 0.0) For a laser source, this is the wavelength of the laser. For a broadband source, this specifies the minimum wavelength in the source.
8. MaxWavelength: (DOUBLE PRECISION, >= MinWavelength) For a Broad Band source, this is the maximum wavelength present in the source.
9. DeltaWavelength: (DOUBLE PRECISION, >0.0) For a Broad Band source, the source is assumed to contain a set of wavelength between MinWAvelength and MaxWavelength, with each wavelength having a power associated with it. This sets the resolution for the sampling of the wavelengths in the source.
10. BeamDiameter: (DOUBLE PRECISION, >0.0) The beam diameter. For a Gaussian beam, this specifies the $1/e^2$ diameter.
11. BeamWaistPosition: (DOUBLE PRECISION, >0.0) Z-coordinate of emitters focus. Beam will be focused at this coordinate.
12. BeamDivergence: (DOUBLE PRECISION) The divergence of the beam, at the focus.
13. PulseDuration: (DOUBLE PRECISION, >0.0) The temporal pulse width.
14. PulsePeriod: (DOUBLE PRECISION, >PulseDuration) The time between the front of multiple pulses.
15. StartTime: (DOUBLE PRECISION, >= 0.0) The time that the emitter comes on and starts pulsing.
16. StopTime: (DOUBLE PRECISION, >StartTime) The time that the emitter turns off.
17. dtMinOn: (DOUBLE PRECISION, >0.0) The minimum time step that the adaptive time step will take when the emitter it turned on. This is what the time step is set to when an emitter is switched on.
18. dtMinOff: (DOUBLE PRECISION, >0.0) The minimum time step that the adaptive time step will take when the emitter it turned off. This is what the time step is set to when an emitter is switched off.
19. StretchOn: (DOUBLE PRECISION, >1.0) The stretch ratio used for the adaptive time step when the emitter is on.
20. StretchOff: (DOUBLE PRECISION, >1.0) The stretch ratio used for the adaptive time step when the emitter is off.

[1] dt starts at a minimum time step and grows with each time step until an emitter state changes. So if an emitter is off at some time, and turns on during the next time step, dt is reset to the minimum value.

12.4 Standard Emitter (*.emitter) Configuration

Standard emitters are indicated in the main configuration file by the StandardEmitter[i] tag. Many of the same configuration options listed in the emitter configuration also apply for the standard emitter configuration. The parameters specific to the standard emitters are listed below.

1. EmitterType: (STRING) The type in the standard emitter configuration is given by a string. Currently supported strings are ”Pade”, ”Hankel”, ”Scattering”.
2. PhotonPacketDensity: (DOUBLE PRECISION, >0.0) For ”Scattering” EmitterType. The number of photons to launch in a 1 cm^2 area.
3. Seed: (INTEGER, >0) For ”Scattering” EmitterType. Seed for initializing random number generator.
4. NoiseReductionFactor: (INTEGER, >= 0) For ”Scattering” EmitterType. Used for smoothing the source term for a scattering emitter near the $r = 0$ axis.
5. zResFactor: (INTEGER, >= 1) For ”Pade” EmitterType. Resolution factor for Pade z axis. Pade propagation grid requires higher resolution than the thermal grid. The propagation grid resolution is based on the thermal grid resolution, multiplied by a resolution factor.
6. rResFactor: (INTEGER, >= 1) For ”Pade” EmitterType. Resolution factor for Pade r axis. Pade propagation grid requires higher resolution than the thermal grid. The propagation grid resolution is based on the thermal grid resolution, multiplied by a resolution factor.
7. OpticsFile: (STRING) For ”Pade” and ”Hankel” EmitterTypes. If given, defines an optics setup for the Gaussian Beam Propagation. Currently, this should be set to ”EYE”, which gives an initial wavefront for the propagation emitter corresponding to the curvature of the eye.
8. P: (INTEGER, >= 0) For ”Hankel” EmitterType. The order of Bessel Functions to use in Hankel propagation. This should be set to zero.
9. numBesselFunctions: (INTEGER, 0 - 4999) For ”Hankel” EmitterType. The number of Bessel Frequencies to use in Hankel transforms.
10. BesselPrecision: (STING, ”single” |”double”) For ”Hankel” EmitterType. Precision to use in Bessel Function evaluation. Double precision evaluation can require much more computational time.

12.5 Layer (*.layer) Configuration

1. LayerType: (INTEGER) An integer that tags the layer.
2. Description: (STRING) A description of the layer.
3. Thickness: (DOUBLE PRECISION, >0.0) The layer thickness
4. Density: (DOUBLE PRECISION, >0.0) The layer density, given in g/cm^3.
5. SpecificHeat: (DOUBLE PRECISION, >0.0) Specific heat of the material.
6. Conductivity: (DOUBLE PRECISION, >0.0) Conductivity of the material.
7. ConvHeatTransRate: (DOUBLE PRECISION, >0.0) The convective heat loss from an air boundary with the material.
8. Emissivity: (DOUBLE PRECISION, 0.0 - 1.0) The emissivity of the material.

9. BloodFlowRate: (DOUBLE PRECISION, >= 0.0) The blood flow rate, given in $g/cm^3 s$, through the material. Setting this to 0.0 turns off blood flow.

10. RefractiveIndex[i]: (STRING) A string specifying the refractive index (n) and change in refractive index ($\frac{dn}{dT}$) for a given wavelength. Values for multiple wavelength can be given by indexing with i. in the STRING, wavelength is given first, then index of refraction, then change in index of refraction with temperature. So, for example, if the index of refraction for the material at 2000 nm is 1.35 and it changes with temperature at a constant rate of 0.1, the entry would be given as:

    RefractiveIndex[0] = "2000 1.35 0.1"

11. Absorption[i]: (STRING) A string specifying the absorption coefficient of the material for a given wavelength. Again, multiple wavelengths can be given yy indexing i.

12. Anisotropy[i]: (STRING) A string specifying the anisotropy of the material at a wavelength. Multiple wavelengths can be given.

13. Scattering[i]: (STRING) A string specifying the scattering coefficient of the material at a wavelength. Multiple wavelengths can be given.

14. Reflectance[i]: (STRING) A string specifying the reflectance of the material at a wavelength. Multiple wavelengths can be given.

15. Temp[i], A[i], Ea[i] (DOUBLE PRECISION) These values specify the normalization constant (A) and the rate process coefficient (E_a) for the damage integral in the material. When calculating the damage integral, the simulation will look for the two temperatures (given in Kelvin) that bracket the current temperature and use the coefficients given for the lower bound. For example, if the following coefficients are listed in the layer configuration file:

    ```
    Temp[0]  =  250
       A[0]  =  10E45
      Ea[0]  =  6.28E10
    Temp[1]  =  300
       A[1]  =  10E99
      Ea[1]  =  3.33E20
    Temp[2]  =  500
       A[2]  =  10E130
      Ea[2]  =  2.08E23
    ```

    and the current temperature is 350 K, then the two coefficients that will be used are $A = 10^{99}$ and $E_a = 3.33^{20}$

```
#KeyValue config file
Dimensions             = VALUE      # INT: number of dimensions in simulation (1 | 2)
SimulationType         = VALUE      # INT: problem (sim) type (0 STD THERMAL, 1 PROPAGATION+THERMAL, 2 Z-SCAN)
AxialGridType          = VALUE      # INT: 0 = UNIFORM, 1 = STRETCHED
Nz                     = VALUE      # INT: number of z grid divisions
zMin                   = VALUE      # DBL: minimum z coordinate
zMax                   = VALUE      # DBL: maximum z coordinate
zStretchRatio          = VALUE      # DBL: (**) z stretch ratio for non-uniform grid
zMinBC                 = VALUE      # INT: z min boundary condition type
                                    #      SINK                     0
                                    #      CONVECTIVE               1
                                    #      RADIATIVE                2
                                    #      CONVECTIVE + RADIATIVE   3
                                    #      EVAPORATIVE              4
                                    #      CONVECTIVE + EVAPORATIVE 5
                                    #      RADIATIVE + EVAPORATIVE  6
                                    #      ALL SURFACE              7
zMaxBC                 = VALUE      # INT: z min boundary condition type
                                    #      SINK                     0
                                    #      CONVECTIVE               1
                                    #      RADIATIVE                2
                                    #      CONVECTIVE + RADIATIVE   3
                                    #      EVAPORATIVE              4
                                    #      CONVECTIVE + EVAPORATIVE 5
                                    #      RADIATIVE + EVAPORATIVE  6
                                    #      ALL SURFACE              7
RadialGridType         = VALUE      # INT: radial grid type (0 = UNIFORM, 1 = STRETCHED)
Nr                     = VALUE      # INT: number of r grid divisions
rMax                   = VALUE      # DBL: Max r coordinate
rStretchRatio          = VALUE      # DBL: (**) r stretch ratio for non-uniform grid
rMaxBC                 = VALUE      # INT: r max boundary condition type (0=SINK; 1=INSULATOR)

TotalSimTime           = VALUE      # DBL: total simulation time
dt                     = VALUE      # DBL: time step for fixed emitter, not for adaptive
dtMax                  = VALUE      # DBL: maximum time step for adaptive emitter (for stability)
TissueBaseTemp         = VALUE      # DBL: tissue baseline temperature
AmbientTemp            = VALUE      # DBL: ambient temperature
RelHumidity            = VALUE      # DBL: relative humidity (0 - 1.0, .6 = 60%)
LogDataFlag            = VALUE      # INT: log data flag (0 = NO, 1=YES)
LogInterval            = VALUE      # INT: log interval in time steps  (10 = LOG 1 in 10)

UserDamageThresh       = VALUE      # DBL: User-defined damage threshold value (damage integral value)
DamageThreshSearchFlag = VALUE      # INT: damage threshold search flag (0 = NO, 1 = YES )
MaxPowerRatio          = VALUE      # DBL: (##) max power ratio for threshold search
MinPowerRatio          = VALUE      # DBL: (##) min power ratio for threshold search
ThreshSearchType       = VALUE      # INT: (##) threshold search type (0=damage,1=dT,2=Les Rad,3=rad dT)
PrimeThreshValue       = VALUE      # DBL: (##) primary threshold value (radius, dT, radius dT)
SecThreshValue         = VALUE      # DBL: (##) secondary threshold value (dT in Thresh Search Type 3)
ConvergThresh          = VALUE      # DBL: (##) convergence threshold for search (e.g. 0.05 = 5% )

Emitter[0]             = "STRING" # STR: emitter file
Layer[0]               = "STRING" # STR: layer filename
Layer[1]               = "STRING" # STR: layer filename

InitialConditionsFlag  = VALUE      # INT: initial conditions flag (0 = none, 1 = read initial conditions file)
InitialConditionsFile  = "STRING" # STR: initial conditions file

PropagationMethod      = VALUE      # INT: propagation method (0=FDBPM, 1=PADE, 2=PADE, Non-Uniform grid)
NumPropSteps           = VALUE      # INT: (++) number of prop steps per grid interval
AperatureCoordinate    = VALUE      # DBL: (++) aperture or screen coordinate [cm]
AperatureRadius        = VALUE      # DBL: (++) aperture radius [cm]
NDiffPattern           = VALUE      # INT: (++) diffraction pattern num points
DiffPatternMaxRadius   = VALUE      # DBL: (++) diffraction pattern max radius
ZScanStartCoord        = VALUE      # DBL: (++) z-scan start coordinate [cm]
ZScanStopCoord         = VALUE      # DBL: (++) z-scan end coordinate [cm]
ZScanStepSize          = VALUE      # DBL: (++) z-scan coordinate step size [cm]
ZScanLensCoord         = VALUE      # DBL: (++) z-scan lens coordinate [cm]
ZScanLensFocalLength   = VALUE      # DBL: (++) z-scan lens focal length [cm]

//-  Notes:

//-  (**) Not used for uniform grid spacing -- code sets to 1.0
//-  (##) Only Used for Threshold Searches
//-  (++) Only Used for Propagation Analysis (and secondary prop grid)

//-  dt is set by emitter when variable timestep emitters are used

//-  See documentation in /doc dirctory for additional parameter information
```

Figure 12.1: Example of the main configuration file

```
#KeyValue emitter configuration
EmitterType           = VALUE    # INT: emitter type   (1 Lsr, 2 BB, 3 Mnstr, 4 Prop, 5 SAR, 6 MCML)
PulseType             = VALUE    # INT: pulse type     (1 Single, 2 Multiple, 3 USER)
FocusType             = VALUE    # INT: focus type     (1 Fixed, 2 Moving [thermal lens])
TimeStepType          = VALUE    # INT: time step type (1 Fixed, 2 Adaptive)
SpectrumType          = VALUE    # INT: for broad band source, type of wavelength spectrum (1 Black Body, 2 USER)
ProfileType           = VALUE    # INT: profile type (1 Gaussian, 2 Tophat, 3 Annular, 4 USER)
PeakPower             = VALUE    # DBL: peak power of emitter [W]
MinWaveLength         = VALUE    # DBL: minum wavelength for broad band source, or wavelength of laser source [1/cm]
MaxWaveLength         = VALUE    # DBL: minum wavelength for broad band source
DeltaWaveLength       = VALUE    # DBL: wavelength step to take for numerical integration
BeamDiameter          = VALUE    # DBL: beam diameter [cm] Note: this is 1/e for gaussian beam profile
BeamWaistPosition     = VALUE    # DBL: position of beam waist
BeamDivergence        = VALUE    # DBL: divergence of beam
PulseDuration         = VALUE    # DBL: duration of one pulse [s]
PulsePeriod           = VALUE    # DBL: period of pulse train [s]
StartTime             = VALUE    # DBL: time that emitter actually comes on [s]
StopTime              = VALUE    # DBL: time that emitter turns off [s]
dtMinOn               = VALUE    # DBL: minimum timestep when emitter is on [s]
dtMinOff              = VALUE    # DBL: minimum timestep when emitter is off [s]
StretchOn             = VALUE    # DBL: timestep stretch ratio when emitter is on
StretchOff            = VALUE    # DBL: timestep stretch ratio when emitter is off
SARFilename           = "STING"      # STR: name of sar file to use
PulseFilename         = "STING"      # STR: name of file with temporal pulse profile
PowerSpectrumFilename = "STING"      # STR: name of power spectrum (power vs wavelength) file
ProfileFilename       = "STING"      # STR: name of beam profile file
FocusFilename         = "STING"      # STR: name of focus vs time file
```

Figure 12.2: Example of the emitter configuration file

```
#KeyValue Standard Emitter configuration

# BaseEmitter Parameters
EmitterType           = "STING"  # STR: emitter type (Scattering, Propagation)
PulseType             = VALUE      # INT: pulse type     (1 Single, 2 Multiple, 3 USER)
SpectrumType          = VALUE      # INT: for broad band source, type of wavelength spectrum (1 Black Body, 2 USER)
ProfileType           = VALUE      # INT: profile type (1 Gaussian, 2 Tophat, 3 Annular, 4 USER)
PeakPower             = VALUE      # DBL: peak power of emitter [W]
MinWaveLength         = VALUE      # DBL: minum wavelength for broad band source, taken to be the wavelength of laser sou
rce [1/cm]
MaxWaveLength         = VALUE      # DBL: minum wavelength for broad band source, taken to be the wavelength of laser sou
rce [1/cm]

FocusType             = VALUE      # INT: focus type     (1 Fixed, 2 Moving [thermal lens])
BeamDiameter          = VALUE      # DBL: beam diameter [cm] Note: this is 1/e for gaussian beam profile
BeamWaistPosition     = VALUE      # DBL: position of beam waist

TimeStepType          = VALUE      # INT: time step type (1 Fixed, 2 Adaptive)
dtMinOn               = VALUE      # DBL: minimum timestep when emitter is on [s]
dtMinOff              = VALUE      # DBL: minimum timestep when emitter is off [s]
StretchOn             = VALUE      # DBL: timestep stretch ratio when emitter is on
StretchOff            = VALUE      # DBL: timestep stretch ratio when emitter is off

PulseDuration         = VALUE      # DBL: duration of one pulse [s]
PulsePeriod           = VALUE      # DBL: period of pulse train [s] so, if period is 1, and duration is .1, this would be
 1 Hz, 10% duty cyle train
StartTime             = VALUE      # DBL: time that emitter actually comes on [s]
StopTime              = VALUE      # DBL: time that emitter turns off [s]

# ScatteringEmitter Parameters
PhotonPacketDensity   = VALUE      # DBL: number of photons per square cm to launch in beam profile
Seed                  = VALUE      # INT: (OPTIONAL) default: 1; seed value to initialize random number generator
NoiseReductionFactor  = VALUE      # INT: (OPTIONAL) default: 0; the number of interier radial rings to average in power
density calculation

# BPEmitter Parameters
#NumPropSteps         = VALUE      # INT: number z steps for propagator to take between thermal z-slices (DEPRECIATED)
zResFactor            = VALUE      # INT: replaces NumPropSteps
rResFactor            = VALUE      # INT: same as zResFactor, but for r direction
OpticsFile            = "STRING" # STR:  Gaussian Beam Propagation Optics Configuration File (could just specify Eye)
# HankelEmitter Parameters
p                     = VALUE      # INT: order of Bessel function to use as propagation basis
numBesselFunctions    = VALUE      # INT: number of pth-order Bessel functions to use in basis
BesselPrecision       = "STRING"      # STR: precision to use in bessel function evaluations (single or double
```

Figure 12.3: Example of the standard emitter configuration file

```
#KeyValue layer configuration file
LayerType             = VALUE    # INT: layer type (see layertypes.txt for list)
Description           = "STRING" # STR: layer description (retinalEP, epidermis, etc)
Thickness             = VALUE    # DBL: layer thickness [cm]
Density               = VALUE    # DBL: layer density [g/cm^3]
SpecificHeat          = VALUE    # DBL: specific heat of layer [J/g/degC]
Conductivity          = VALUE    # DBL: conductivity [J/s/cm/degC]
ConvHeatTransRate     = VALUE    # DBL: convective heat transfer rate [J/s/cm2/degc]
Emissivity            = VALUE    # DBL: emissivity of tissue (0 - 1)
BloodFlowRate         = VALUE    # DBL: blood flow rate [g/(cm^3*s)]

#Refractive index and dn/dt vs wavelength   - STR ("DBL DBL DBL"): --> "\lambda n dn/dt"
RefractiveIndex[0]    = "VALUE VALUE VALUE"
RefractiveIndex[1]    = "VALUE VALUE VALUE"

#Absorption Coefficient vs wavelength       - STR ("DBL DBL"): --> "\lambda \mu_{a}" [1/cm]
Absorption[0]         = "VALUE   VALUE"
Absorption[1]         = "VALUE   VALUE"

#Scattering Anisotropy vs wavelength        - STR ("DBL DBL"): scattering angle cos by wavelength "\lambda \cos(\theta)"
Anisotropy[0]         = "VALUE   VALUE"
Anisotropy[1]         = "VALUE   VALUE"

#Scattering Coefficients vs wavelengt       - STR ("DBL DBL"): scattering coeffiecients by wavelength "\lambda \mu{s}"
Scattering[0]         = "VALUE   VALUE"
Scattering[1]         = "VALUE   VALUE"

#Reflectance values vs wavelentgh           - STR ("DBL DBL"): 0 - 1
Reflectance[0]        = "VALUE   VALUE"
Reflectance[1]        = "VALUE   VALUE"

Temp[0]               = VALUE    # DBL: temperature for rate ceofficients
   A[0]               = VALUE    # DBL: A rate coef. [1/s]
  Ea[0]               = VALUE    # DBL: Ea rate coef. [J/mole]
Temp[1]               = VALUE    # DBL: temperature for rate ceofficients
   A[1]               = VALUE    # DBL: A rate coef. [1/s]
  Ea[1]               = VALUE    # DBL: Ea rate coef. [J/mole]
```

Figure 12.4: Example of the layer configuration file

Acknowledgements

This work was sponsored by the USAF Research Laboratory and the Air Force Office of Scientific Research. C.D. Clark, L. J. Irvin and G.D. Buffington acknowledge the support of USAF Research Laboratory Contract F44624-02-D7003. P.D.S.Maseberg and J. Stolarski acknowledge the support of the Air Force Research Laboratory Human Effectiveness Directorate Consortium Research Fellows Program. The ideas and opinions presented here are those of the authors and not those of the US Air Force or the Department of Defense. The authors wish to thank Bo Chen of the University of Texas at Austin for useful discussions in the refinement of thermal models and boundary condition parameters, and for providing validation data.

THIS PAGE INTENTIONALLY LEFT BLANK

References

[1] Ronald G. Hadley. Wide-angle beam propagation using padé approximant operators. *OPTICS LETTERS*, 17(20):1426–1428, 1992.

[2] Sami T. Hendow and Sami A. Shakir. Recursive numerical solution for nonlinear wave propagation in fibers and cylindrically symmetric systems. *Applied Optics*, 25(11):1759–1764, 1986.

[3] M. A. Mainster, T. J. White, J. H. Tips, and P. W. Wilson. Transient thermal behavior in biological systems. *Bulletin of Math Biophysics*, 32:303–314, 1970.

[4] Peter W. Milonni and Joseph H. Eberly. *Lasers*. Wiley-Interscience, first edition, 1998.

[5] M. Necati Ozisik. *Boundary value problems of heat conduction*. International Textbook Company, 1968.

[6] M. Necati Ozisik. *Heat Conduction*. John Wiley & Sons, 1980.

[7] M. Necati Ozisik. *Finite Difference Methods in Heat Transfer*. CRC Press, 1994.

[8] D. W. Peaceman and H. H. Rachford. The numerical solution of parabolic and elliptic differential equations. *Journal of the Society for Industrial and Applied Mathematics*, 3:28–41, 1955.

[9] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C++, The Art of Scientific Computing*. Cambridge University Press, New York, second edition, 2002.

[10] G. D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*. Oxford Applied Mathematics and Computing Science Series. Oxford University Press, Oxford, thrid edition, 1998.

[11] A. Taflove and S. C. Hagness. *Computational Electrodynamics - The Finite-Difference Time-Domain method*. Artech House, Norwood, MA, second edition, 2000.

[12] Lihong Wang, Steven L. Jacques, and Liqiong Zheng. Mcml - monte carlo modeling of light transport in multi-layered tissues. *Computer Methods and Programs in Biomedicine*, 47:131–146, 1995.

[13] A. J. Welch and M. J. C. van Gemert. *Optical-Thermal Response of Laser-Irradiated Tissue*. Lasers, Photonics, and Electro-Optics. Plenum Press, New York, first edition, 1995.

[14] K. S. Yee. Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions Antennas and Propagation*, 14:302–307, 1966.